https://medium.com/@GeoffreyGordonAshbrook/lets-test-models-and-let-s-do-tasks-84777f80eb99

Let's Test Models and Let's Do Tasks

2024.03.11-25 G.G.Ashbrook

What if there were a way to enter sets of tests (questions and answers), including either a standard benchmark-test or your own question-answer-sets made by you, into a "system" along with the names of some AI models, and then the system would (as you specify, for example offset and range, randomizing the order of the multiple-choice options, letting you specify how many guesses and retries a model would be allowed, how much it can work on selecting the best final answer, doing code-writing tests, multiple-guess, or open answer tests) run each model through each set of test-questions, record the output and associated data (log errors, count retries, time the tests) in tabular form (like a spreadsheet), including possible error analysis for each incorrect answer, enter the test scores onto a separate tally-sheet-by-model comparing the performance of each models on your specified tests, generate HTML summaries, and where this all works for both locally run .gguf models and for cloud-api models, comparing either different models to each-other or even the same model for stability and performance with different configurations on the same inputs? The tools presented here are an initial version of such a tool-set, aiming to be part of much more.

A general walkthrough and discussions of topics related to practical use are included here.

Repository: See the readme.md for more detailed run instructions.
- https://github.com/stemnetbenchmarks/lets_test_models

Six Steps to Get Started:
1. (Optional: install jan)
2. Download some .gguf models (easy with jan)
3. Install llama.cpp
4. Install python, if not already installed
5. Clone repo for 'Let's Test Models'
6. Run python do_task.py

- For local .gguf models, no python environment or packages are needed
- cloud-mode works with Mistral and Antropic, set up an .env and use python-dotenv

Goals, Means, Methods

We need tools that:
- are broadly Accessible
- work on most laptop desktops, with standard specs,
- transparent low-level development tools oriented towards projects
- at least an attempt to have a broad or configurable set of inputs
- are shareable, externalizable, results and outputs that people can meaningfully compare
- support the development and testing of models for specific tasks, especially small and locally run models
- improving training sets broadly including general and specific tests for ai-ml and for people (and any other social participants).
- handle standardized multiple choice tests and also open-answer tasks
- handle code-generation and output management tests and tasks (ToDo)
- can perform project-architecture role tasks (such as systematic translation in a context of specific input and output specs
- facilitate essential formalities such as retry, error checking, timeout, throttle and glitch guarding, etc.
- to give people a hands-on way of seeing what AI can really do, both to make better every-day decisions about what to use it for, and to orient and tether their mind and imagination to reality (where desired) so they are not hoping or fearing too far in fictional directions.
- to better interpret claims made in papers and in the news.
- to 'democratize' testing, from running tests to making tests
- supporting STEM-Net Benchmarks: to help develop better tests and curricula for people (Homo-sapiens humans) and for AI (and whatever other categories) including tools for making training-sets, better data-augmentation tools, etc.
- in line with Object-Relationship-Space frameworks: To move toward 'ai architectures' and project-architectures. Doing tasks with STEM-tools is really not as simple as people make it out to be. This will be a long journey that in many ways we have yet to start.
- support with standard test sets (e.g. classic Winograd schema test support currently added).

Walkthrough:

You just thought of a new question that you would like to put to several models to see how good a job each can do at selecting the correct answer (or, alternately, you thought of an open-answer question and you wish to see what each model decides is the best answer without any options for answers provided by you). You might even want to try both, just to see what happens.

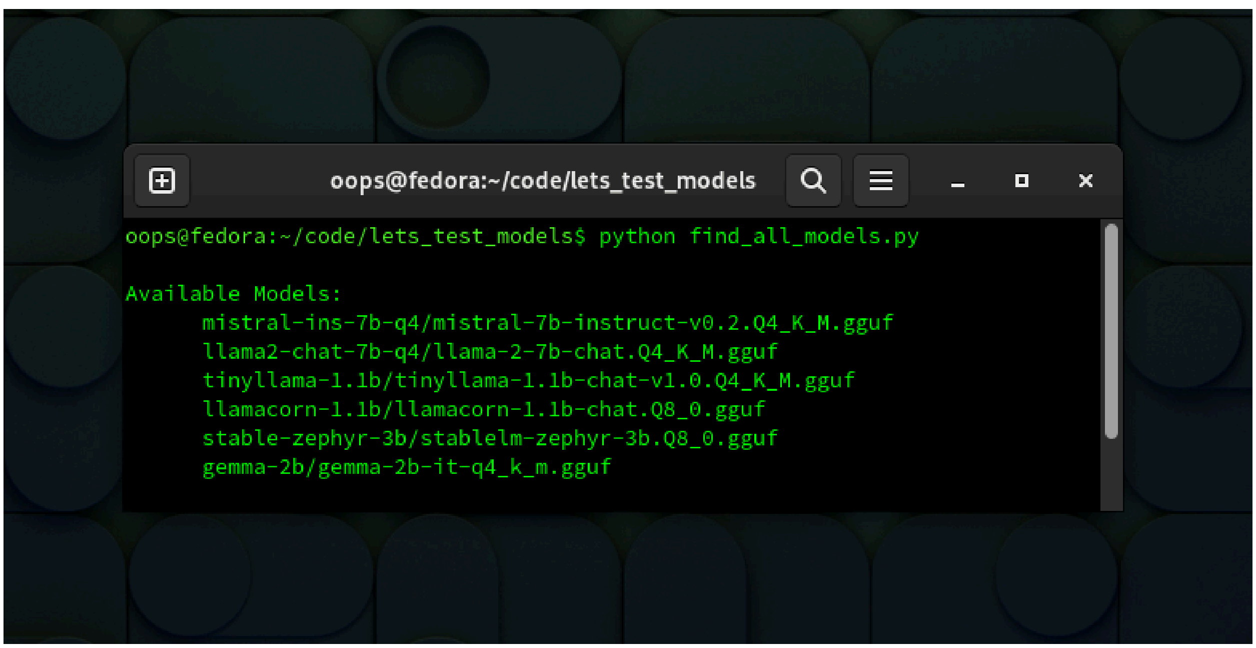
After making sure python is installed, installing llamacpp, and cloning the code repository for this project, you create the tests that you dreamed up using these tools:

1. You open up "make_a_test.py", or "make_a_code_test.py", in any text editor you prefer.
2. You type in your questions and other data.
3. You run the test-maker in a terminal with a command like: python make_a_test.py
4. You move the test files that you just made into the /ai_task_files/ folder (or 'directory').

Now you have your standardized test file (which you can also now easily share with other people).

Next you decide how and where you want to run the test (or tests). Many of these choices may be the same each time and can be left to a 'default' answer.

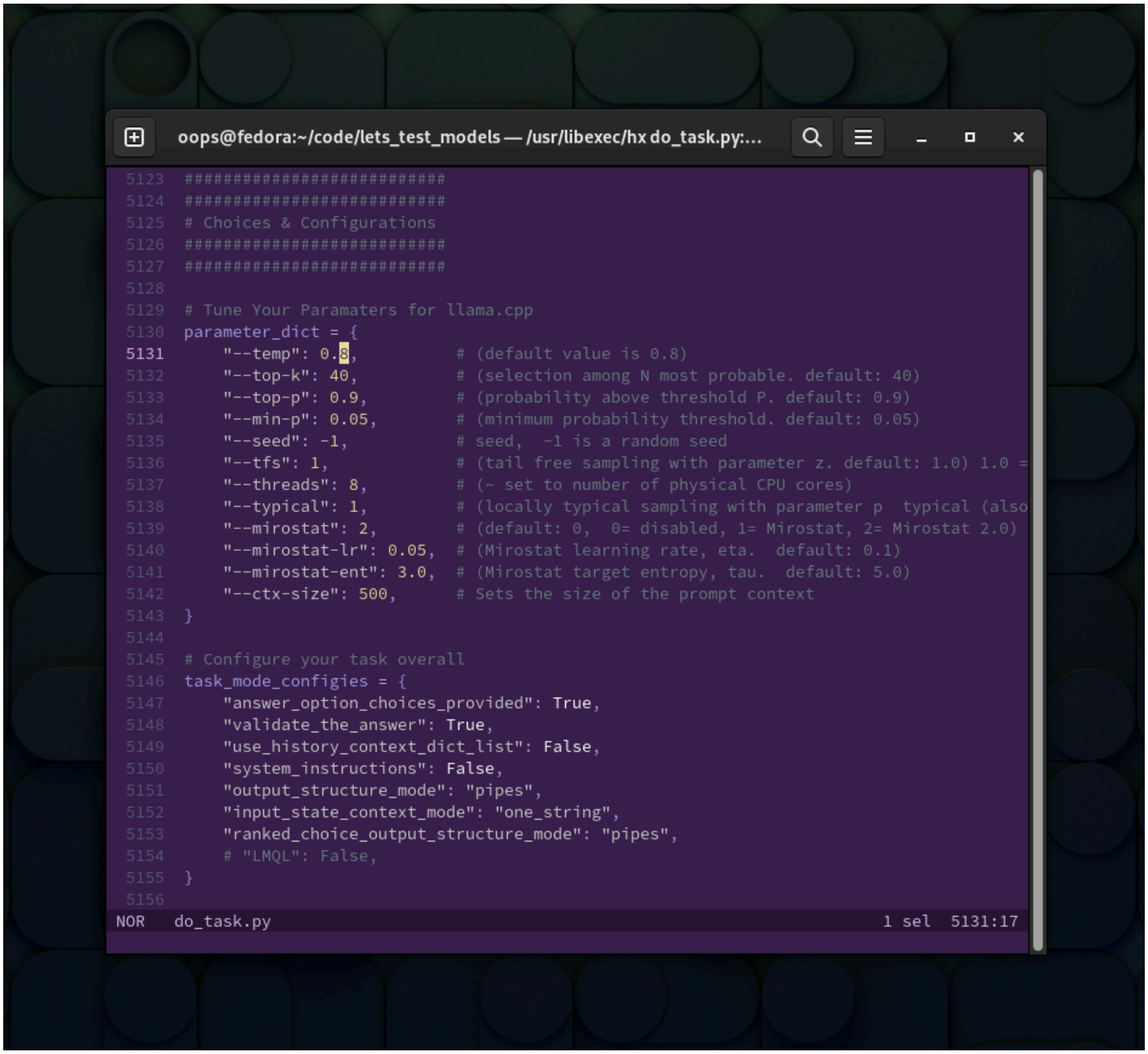
For example, to see what models you can currently use for tasks locally, you run: python find_all_models.py



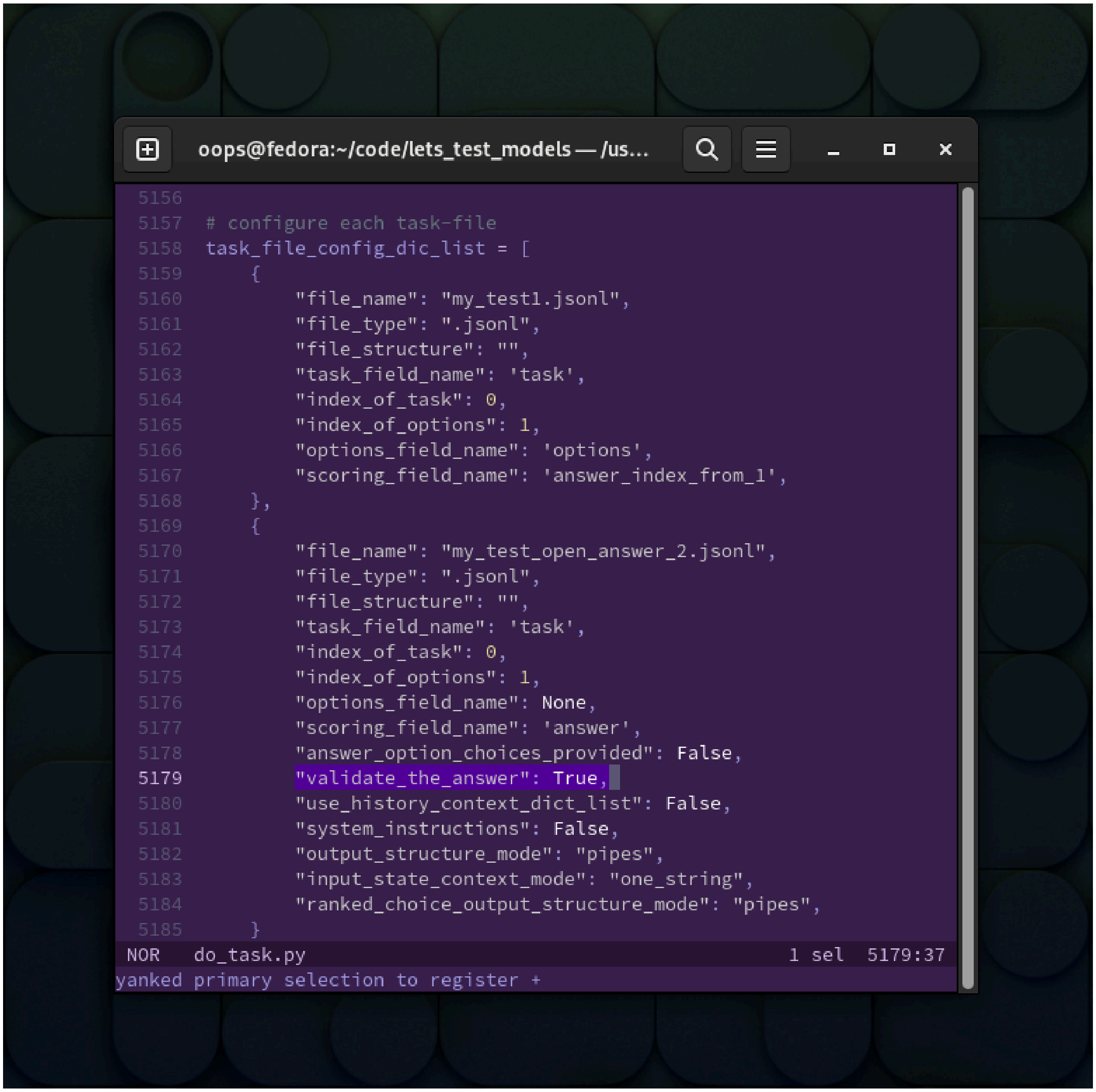
As long as there is only one version of a given model, you do not have to specify the whole long name. But where you do have a few specific versions of a model, you can use the full but less easily pronounceable name.

Most of the lower level configurations you may not change often, but you can change them if you wish.

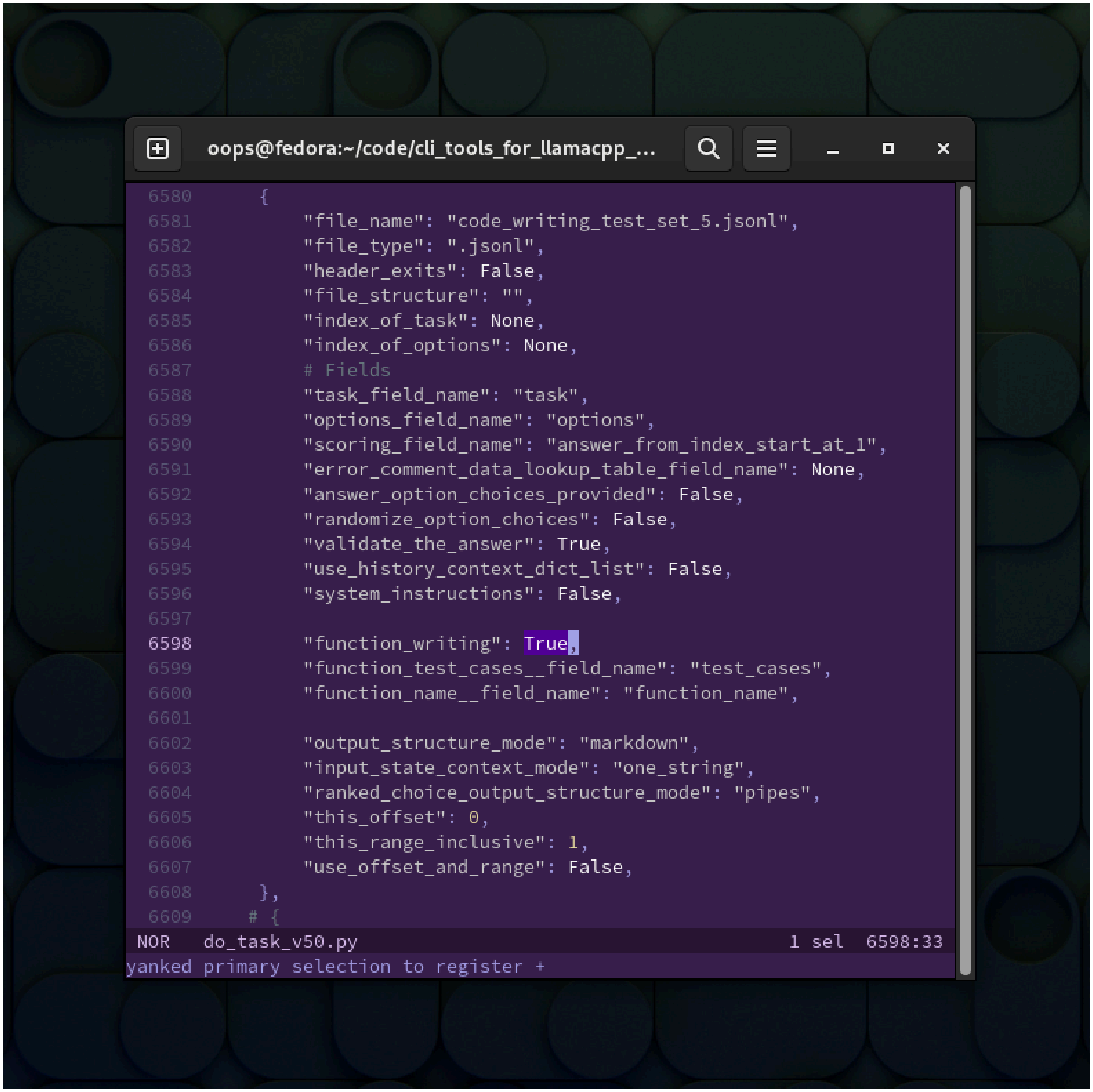
So you open up do_task.py and check the configurations at the bottom of the file.



As these are 'standardized tests,' in many cases you might just run the tests without any changes to configuration. But let's say you want to run this new test that you just had a brainwave about on a few specific ai models, and you want to try to score the answer on the open-answer test, so you want to make sure that is turned on. (See: validate_the_answer is set to: True)



Or, similarly, if you are testing the ai-model's ability to write code, you set the 'function_writing' key to the value of: True.



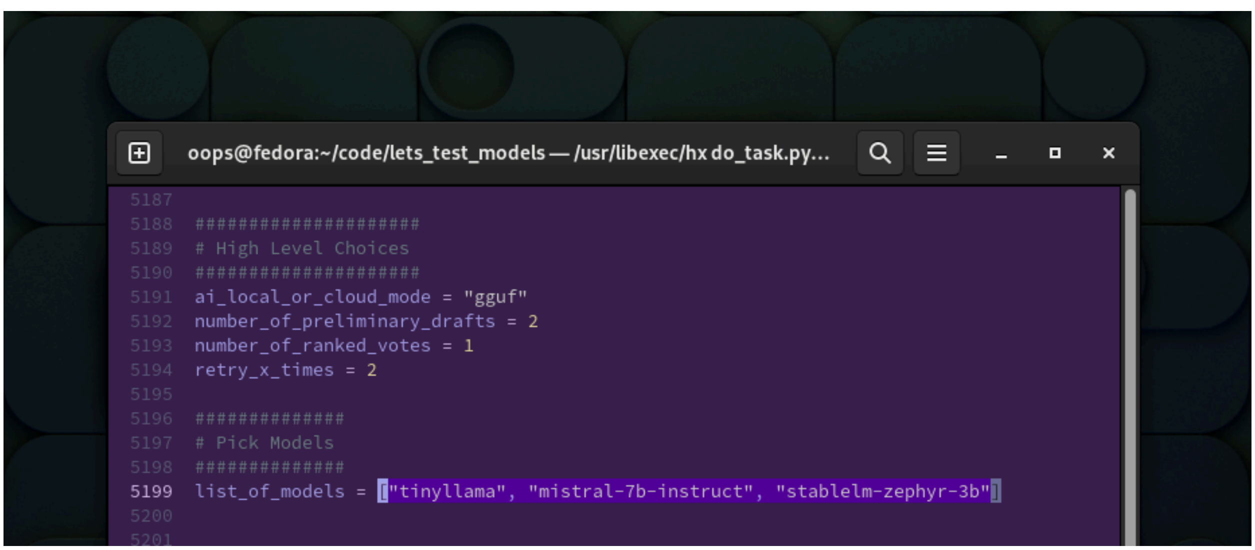

```
    {
        "file_name": "code_writing_test_set_5.jsonl",
        "file_type": ".jsonl",
        "header_exits": False,
        "file_structure": "",
        "index_of_task": None,
        "index_of_options": None,
        # Fields
        "task_field_name": "task",
        "options_field_name": "options",
        "scoring_field_name": "answer_from_index_start_at_1",
        "error_comment_data_lookup_table_field_name": None,
        "answer_option_choices_provided": False,
        "randomize_option_choices": False,
        "validate_the_answer": True,
        "use_history_context_dict_list": False,
        "system_instructions": False,

        "function_writing": True,
        "function_test_cases__field_name": "test_cases",
        "function_name__field_name": "function_name",

        "output_structure_mode": "markdown",
        "input_state_context_mode": "one_string",
        "ranked_choice_output_structure_mode": "pipes",
        "this_offset": 0,
        "this_range_inclusive": 1,
        "use_offset_and_range": False,
    },
    # {
```

You can also use options such as offset and range, for when you want to run a portion of a test file but not all the questions (staring at offset, and running for range). "randomize_option_choices" lets you mix up the order of the answers, which is at least some protection against the model having been trained on this exact test and inadvertently having learned that the first answer provided is correct for a given question (e.g. you can re-test the same model on randomized choice versions of the same test to see if that makes any difference in the performance of the model, ideally the order of the choices should not matter).

One configuration dictionary for each test set:
Because each test file (each set of questions and answers) may have a substantially different structure, each incoming task file can have its own configuration. This is not needed if you are running a task across already standardized structures such as translating json files, but tests are not all the same. As your projects will be open ended, there is no standardized format for what you are allowed to do; you can do anything that code can do.

Notice here that while you set overall choices, in some cases for a specific test you will need to override a default setting. For example, some tests may not have provided correct answers. You might just want to see what the model says. Or some test might have non-simple multiple correct answers (select all that apply). You need the flexibility to account for any possible variation in a test's structure.

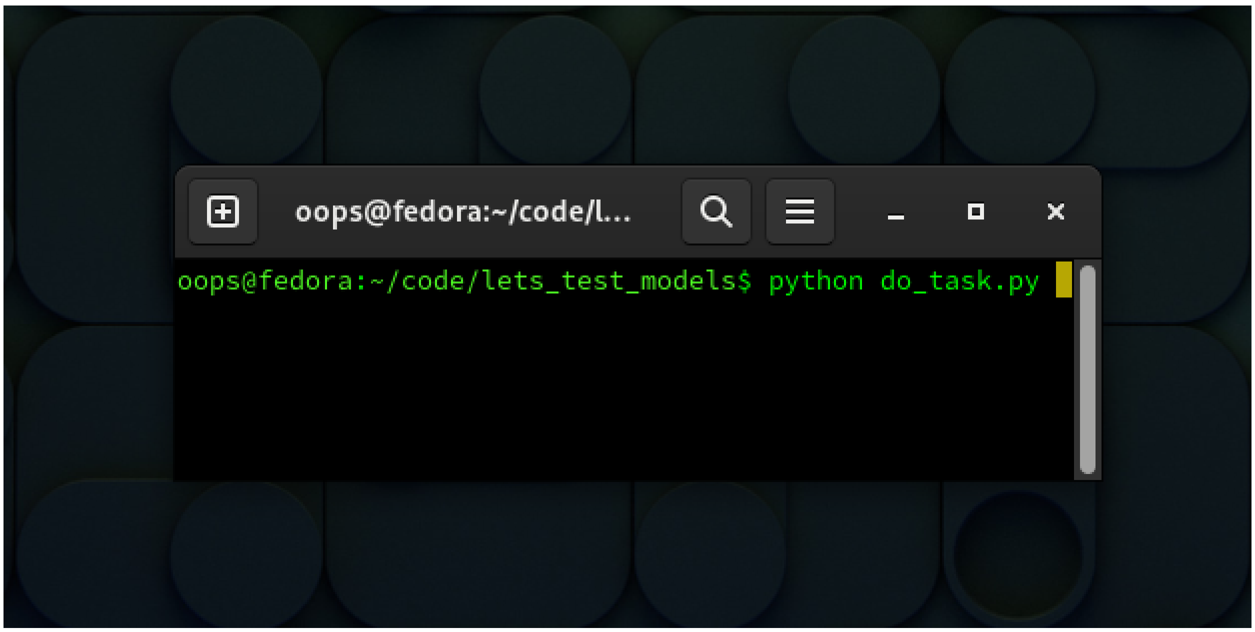
Finally you can select your higher-level choices: How many draft-answers do you want each ai model to make for each task (or test question in this case), how many best-answer-evaluations do you want each ai model to make for its final best answer? How many times do you want to retry in the case of myriad possible failed-try issues? And, possibly the only thing you will need to select in many cases, which models to test!?



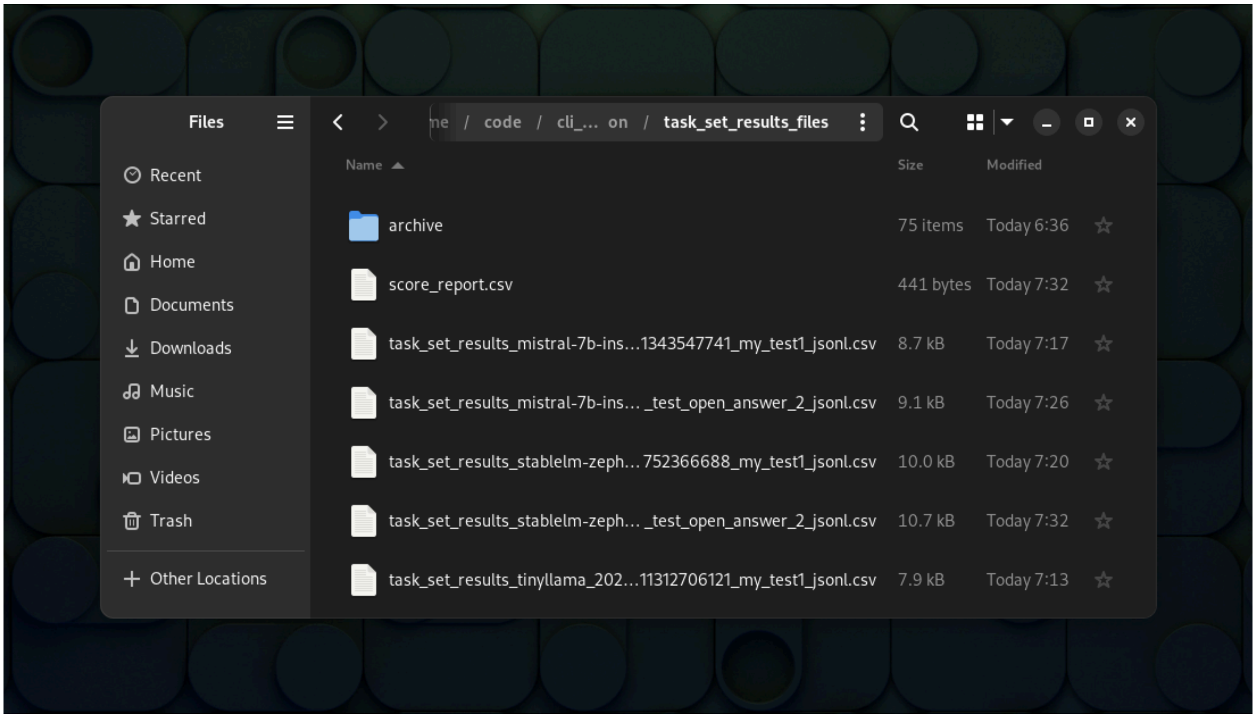
Note here where it says 'ai_local_or_cloud_mode = "gguf" ,' this is one place where you will make a change if you want to use a cloud api such as Mistral, Anthropic(Claud) or OpenAI(GPT), changing "gguf" to "cloud" and there are a few other steps to set up api-use (making a python environment for packages for using a cloud API, and uncommenting the code for importing those, and set up your .env secret "environment variables" file and uncommenting python dot-env to get your keys safely out).

(Note: When using a paid service, you will need to use your own key and you will be charged so be careful not to start an expensive task or big set of tests if you do not want to. Off-set and range may be useful here, to run just a few questions.)

Then, when you are ready to go, you run the command: python do_task.py



The system runs all the questions past all the models, collects the results, and makes your reports. This might take quite a few minutes, especially for longer tasks on slower models or slower hardware. The answer file for each task for each model and the overall tally file should appear automatically in the task_set_results_files folder.



Then you can begin the real job, and the real fun, of collaboratively digging into, discussing, scrutinizing, and acting upon those data and sharing your results and thoughts.

A score tally can be made any time using any read-able answer/task_results files (and will be auto-generated after a run as well). For Example: .csv

```
"percent","model","score","time_stamp"
"0.0","mistral-7b-instruct","0 / 2","2024-03-23-20:51:16578539"
"57.14285714285714","mistral-7b-instruct","4 / 7","2024-03-23-21:36:19601255"
```

Likewise, an html summary for a single report file or all report files can be made any time using html_all_reports_summary_from_csv.py and html_tally_score.py (and will be auto-generated after a run as well). A bit easier on the eyes.

CSV Summary

Score	Selected Option	Correct Option	Task Failure Comment	Name of Model	Task File	Task from Instructions	Error Log	Duration of Single Task	Readable Timestamp
0	3	1	This is incorrect because it assumes that the drying time increases linearly with the number of shirts. However, the sun can dry all 10 shirts simultaneously, so the drying time remains the same as drying one shirt.	mistral-7b-instruct	error_explained_test_1.jsonl	If it takes 30 minutes to dry one shirt in the sun(comma) how long does it take to dry 10 shirts in the sun simultaneously?		4_min__25.8_sec	ymd_2024-03-23
1	1	1		mistral-7b-instruct	error_explained_test_1.jsonl	If it takes 30 minutes to dry one shirt in the sun(comma) how long does it take to dry 10 shirts in the sun simultaneously?		3_min__16.7_sec	ymd_2024-03-23
0	2	1	This is incorrect because cooking multiple pancakes simultaneously does not increase the cooking time. The griddle can cook all 8 pancakes at the same time.	mistral-7b-instruct	error_explained_test_1.jsonl	If it takes 5 minutes to cook one pancake on a griddle(comma) how long does it take to cook 8 pancakes simultaneously using the same griddle?		3_min__32.2_sec	ymd_2024-03-23

```
Looks like cloud api mistral-tiny got... 1/3: Not bad.
```

1	1	1		mistral-tiny	error_explained_test_1.jsonl	If it takes 30 minutes to dry one shirt in the sun(comma) how long does it take to dry 10 shirts in the sun simultaneously?		0_min__6.5_sec	ymd_2024-03-24
0	3	1	This is incorrect because it assumes that the cooking time increases linearly with the number of pancakes. However, the griddle can cook all 8 pancakes simultaneously, so the cooking time remains the same as cooking one pancake.	mistral-tiny	error_explained_test_1.jsonl	If it takes 5 minutes to cook one pancake on a griddle(comma) how long does it take to cook 8 pancakes simultaneously using the same griddle?		0_min__9.6_sec	ymd_2024-03-24
0	3	1	This is incorrect because it assumes that the washing time increases linearly with the number of loads. However, the washing machine can wash all 4 loads simultaneously, so the washing time remains the same as washing one load.	mistral-tiny	error_explained_test_1.jsonl	If it takes 30 minutes to wash one load of laundry in the washing machine(comma) how long does it take to wash 4 loads of laundry simultaneously using the same washing machine at the same setting(comma) assuming the load fits in?		0_min__7.3_sec	ymd_2024-03-24

```
How about Anthropic's Claud-2 and Claud-3?
```

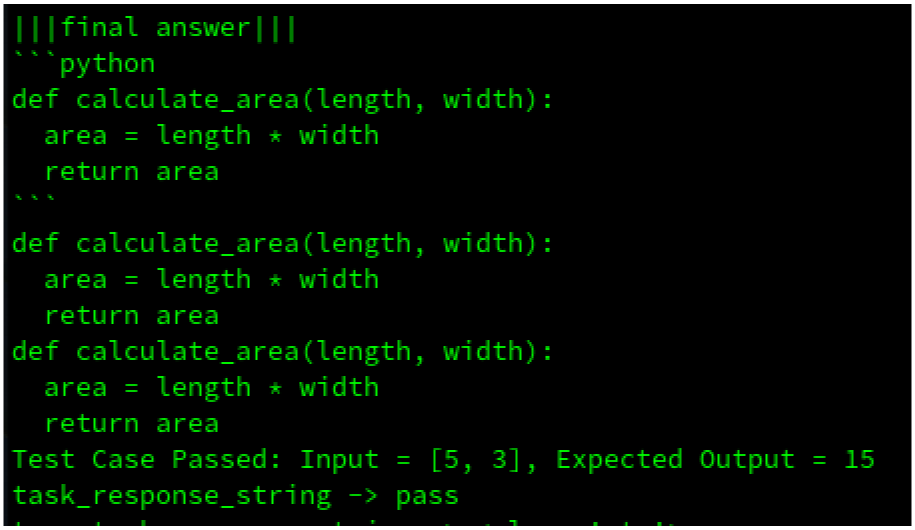
Score Tally Summary

Percent	Model	Task File	Score	Timestamp
83.33333333333334	claude-3-opus-20240229	10 / 12	winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl	2024-03-24-23:14:14006367
75.0	claude-2.1	9 / 12	winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl	2024-03-24-23:14:14006747

Score Tally Summary

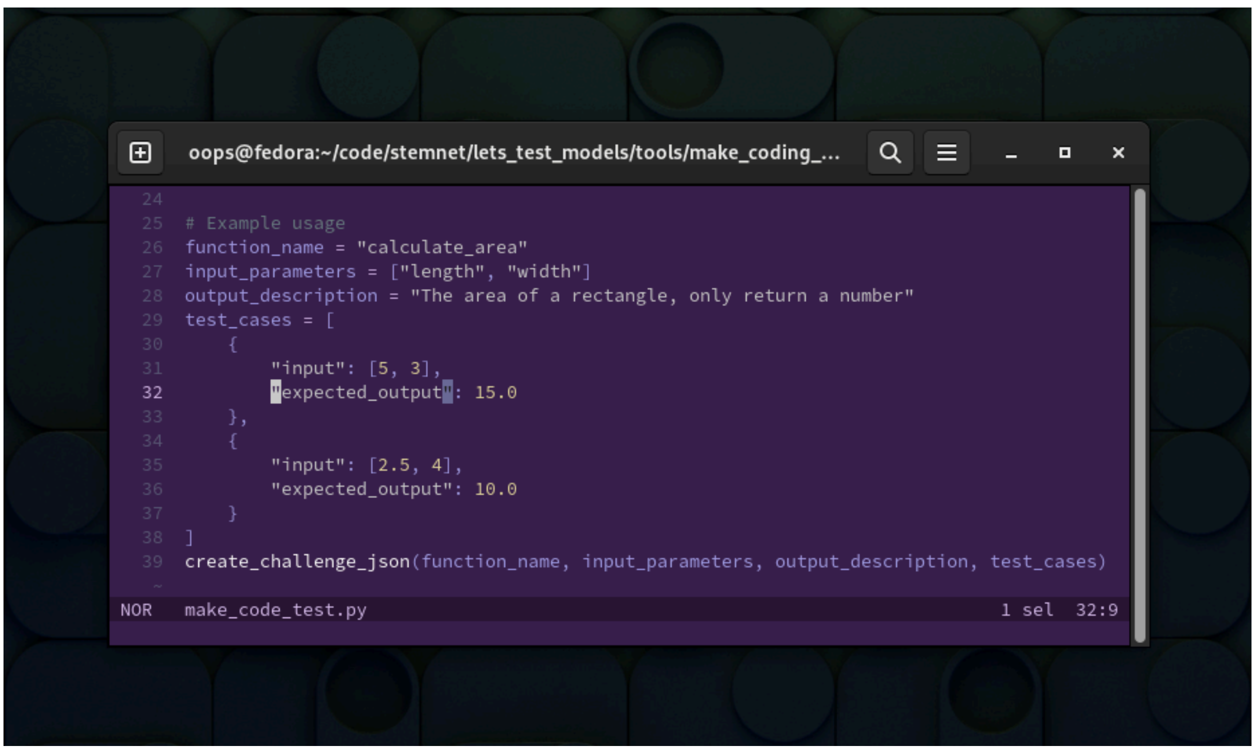
Percent	Model	Task File	Score	Timestamp
36.53846153846153	mistral-7b-instruct	error_explained_test_1.jsonl, error_explained_test_1.jsonl, error_explained_test_1.jsonl, error_explained_test_1.jsonl, error_explained_test_1.jsonl, error_explained_test_1.jsonl, error_explained_test_1.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, my_test1.jsonl, my_test1.jsonl, my_test_open_answer_2.jsonl, my_test_open_answer_2.jsonl, error_explained_test_1.jsonl, error_explained_test_1.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl	19 / 36	2024-03-24-16:06:17174154
7.6923076923076925	stable-zephyr-3b	my_test1.jsonl, my_test1.jsonl, my_test_open_answer_2.jsonl, my_test_open_answer_2.jsonl, error_explained_test_1.jsonl, error_explained_test_1.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl, winograd_schemas_test_file.jsonl	4 / 16	2024-03-24-16:06:17174475

If you want to compare models' abilities to write python code in the format of a coding challenge where you specify the input and output specifications and the code-tests, the ai writes the functions and your tests are run.



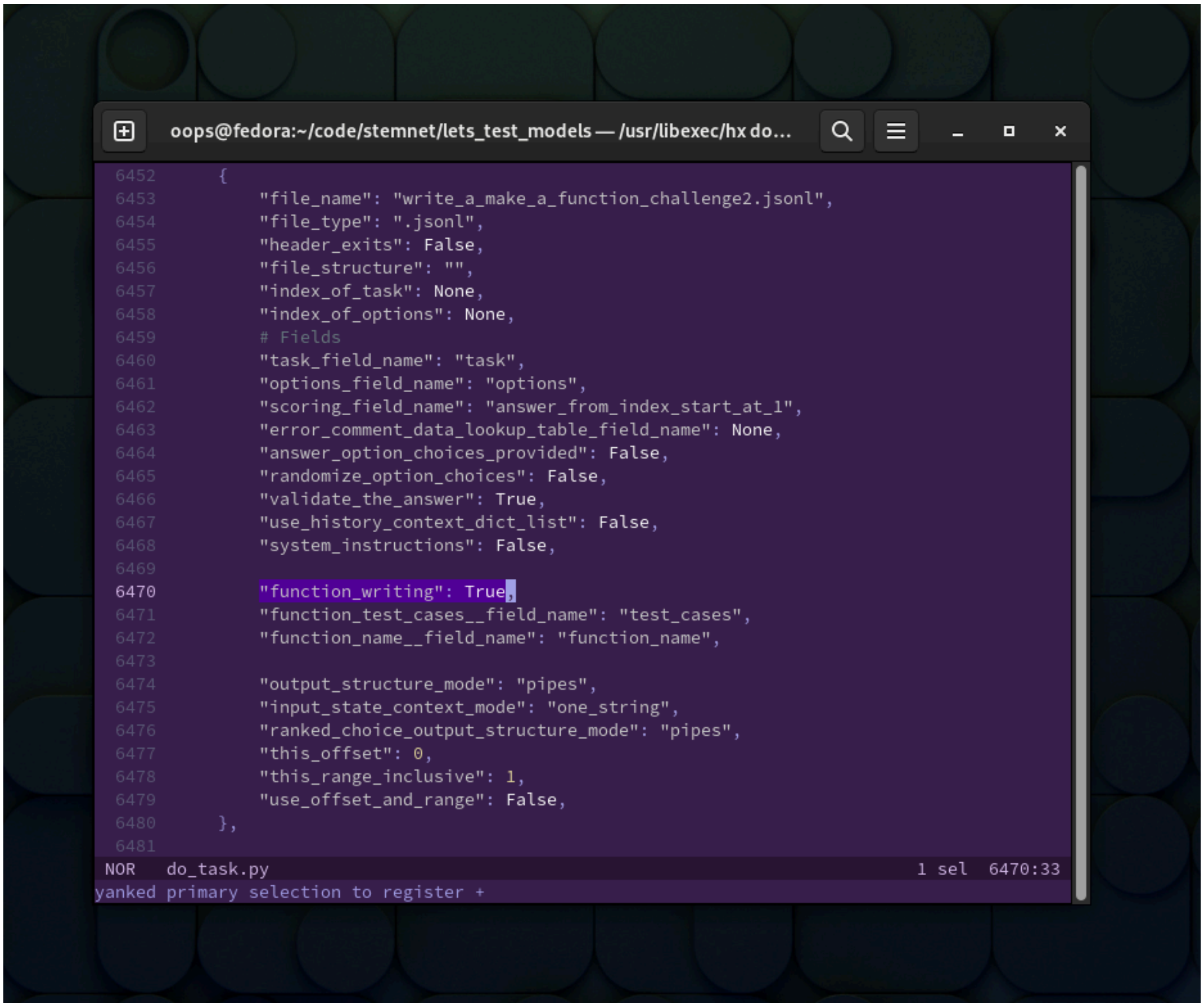
````
|||final answer|||
```python
def calculate_area(length, width):
  area = length * width
  return area
```
def calculate_area(length, width):
  area = length * width
  return area
def calculate_area(length, width):
  area = length * width
  return area
Test Case Passed: Input = [5, 3], Expected Output = 15
task_response_string -> pass
````

You can create your own sets of coding challenges with the make_coding_tests tools in the tools directory. (See example.)




```python

# Example usage
function_name = "calculate_area"
input_parameters = ["length", "width"]
output_description = "The area of a rectangle, only return a number"
test_cases = [
    {
        "input": [5, 3],
        "expected_output": 15.0
    },
    {
        "input": [2.5, 4],
        "expected_output": 10.0
    }
]
create_challenge_json(function_name, input_parameters, output_description, test_cases)
```


Set your configuration file for function_writing. (See example.)



And the results will be scored as pass/fail if the code passes all the test cases (unit tests) you create for the challenge.

CSV Summary

Score	Selected Option	Correct Option	Task Failure Comment	Name of Model	Task File	Task from Instructions	Retries	Error Log	Duration of Single Task	Readable Timestamp
1	pass	pass		llamacorn	write_a_make_a_function_challenge2.jsonl	Write a python function called calculate_area()(comma) such that given input(s) are (left square bracket)(single quote or apostrophe)length(single quote or apostrophe)(comma) (single quote or apostrophe)width(single quote or apostrophe)(right square bracket) and the output is The area of a rectangle(comma) only a number..	0		0_min__16.0_sec	ymd_2024-03-31
0	None	pass		tinyllama	write_a_make_a_function_challenge2.jsonl	Write a python function called calculate_area()(comma) such that given input(s) are (left square bracket)(single quote or apostrophe)length(single quote or apostrophe)(comma) (single quote or apostrophe)width(single quote or apostrophe)(right square bracket) and the output is The area of a rectangle(comma) only a number..	24	(left square bracket)(single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(right square bracket)	1_min__14.9_sec	ymd_2024-03-31
1	pass	pass		stablelm-zephyr-3b	write_a_make_a_function_challenge2.jsonl	Write a python function called calculate_area()(comma) such that given input(s) are (left square bracket)(single quote or apostrophe)length(single quote or apostrophe)(comma) (single quote or apostrophe)width(single quote or apostrophe)(right square bracket) and the output is The area of a rectangle(comma) only a number..	12	(left square bracket)(single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(comma) (single quote or apostrophe)(single quote or apostrophe)(right square bracket)	3_min__23.3_sec	ymd_2024-03-31

A short overview of Topics in Testing Code Writing Abilities of Generative Models:

Here are a few topics on the subject of testing code writing abilities of generative models, hopefully to be covered elsewhere in more depth.

Here are three contexts, none of which are 'the one true' context for talking about code, and all of which cover parts of the subject.

1. array of tools:
- using steps (or not)
- using code (or not)
- using an ALU (or not)
- having a specified output
- having a unit-testable output
- generating a process report

2. Tactical vs. Strategic
In short, computers broadly have tended to be best at pre-programmed calculation tasks. This pattern operates on various levels from the speed of optimized calculation to higher level 'Kasparov Event Horizon' described by Gary Kasparov (not using that exact term) over his many decades of working with chess software developers (see his book 'Deep Thinking') and a phenomena that may be easily recognizable to anyone who has played many games of iGo or chess against various software-opponents: software rarely makes tactical calculation mistakes, it won't panic and get backwards the order of which of two pieces should move first in a simple situation. But overall deeper strategy is usually much harder, and you can defeat software by staying ahead of the software strategically where it is near-sighted and avoiding tactical-calculation competitions where the machine usually has the advantage.

This also may figure in ai writing code. AI may be able to write small functions (tactical coding) quickly and well, but when the task turns to the strategic choices of how to use or design function in combination the ability (just as in iGo or chess) can rapidly fall and fail.

3. Function-Library Management:
- creating functions
- removing functions
- selecting and searching for functions
- modifying functions
- recombining functions
- combining different types of functions

There are many tasks and tests that can involve writing code, this is just a minimal point of entry that may still be useful for putting your own tests to models rather than needing to rely upon the claims of others.

As a last walk-though summary note, I also recommend using the call_llamacapp.py file directly when you want to see how one specific model reacts to one specific prompt (or type of prompt). This is useful when you are configuring a new test and you want a more direct way to get that specific information. For example, it takes time to run many models and many test questions or sets of questions. This can take even longer if one model is unable to take the test, slowly spewing nonsense and triggering every allowed retake option. Calling call_llamacapp.py directly on that model is very useful here, removing the need to run such a model through an entire testing run in order to get any data about how that model performs. call_llamacapp.py is also what the test system itself uses, so it is a closer-test than trying to run a similar-ish test question through some other version or way of running the model such as a cloud API or another interface (ollama, huggingface, llamaindex, langchain, other versions and wrappers for llamacpp, nitro, groq, tensorflow, pytorch, etc.). You might think of this as a micro-test, without many of the automations and their configurations.

Principle Files

For many people the only file you may need is do_task.py to run your tests. But if you want to try other projects there are sample batch-translation colabs and an incremental 'crawler' translator.py, and 'call_llamacpp' will give you an easy ability to test out single prompts which can very useful if you want to experiment with out the system reacts to a specific question or prompt, but you don't want to run a whole batch of tests. call_llamacpp.py is what do_task.py uses to interface with the model at the one-question level.

1. call_llamacpp.py (utility)
2. do_task.py (tests and template for tasks)
3. json_translator(...).py / batch_translator.ipynb (specific task)

What are these tools, and how can we use them?

This is part of a set of tools based on and using the original cli (command line interface) version of llama.cpp by Georgi Gerganov
https://github.com/ggerganov/llama.cpp
(as with so many other things the name can be confusing because the term 'llama cpp' has proliferated across many (related and unrelated) tools).

It is often important to use, test, and develop a locally run model but often only the most popular cloud api services are known or available. The aim here is to provide some of the formal steps and tools needed to use locally run .gguf formatted models (of which there are many on Huggingface) in the same way that many application use (and are stuck in relying solely upon) api interfaces often (but not always) for closed-source models.

I recommend using the tools presented here alongside the laudable "Jan" platform/framework/thing https://github.com/janhq/jan . (Again, naming is difficult, beware that many products share a similar name.)

The tools presented here do not at all rely upon Jan or in any way directly connect to Jan. You do not have to use Jan at all, but the purposes and functions of Jan are arguably highly complimentary. And a comparison with Jan may even help to illustrate the purposes and uses of the tools presented here. For example, could you just use Jan or llama.cpp itself for your project instead of the tools presented here? Depending on the details, possibly you should.

Let's try to look at this in terms of levels from very-low-level to very-high-level. Llama.cpp is very low-level and it is generally used and integrated-with by software under the hood and not by ordinary users interacting with ai-models. It has no graphic user interface at all, and the command line interface is rather technical, inputting and outputting things that have more to do with software configurations and less to do with human-level tasks (In other words, not human language or anything that makes sense in day to day projects). At the other end of the spectrum, Jan (which is also built around llama.cpp) is a wonderful no-code higher-level graphic user interface that operates at a human-language and concept level. Jan makes it extremely easy to download models from hugging-face with one button click and then to chat with those models with just one more button click. AND you can manually download any of the thousands of .gguf models on huggingface, and merely drop them into their own folder in the jan/models folder, and poof, just like that they appear in the list next time you start Jan. It saves an enormous amount of time and makes testing and comparing models (with simple chat prompts) exponentially more practical and convenient. And it is all open source, all local. You can use it during a power-outage on a laptop with no internet (as I actually had to do during a power and internet outage recently to look up historical timeline data about the history of unix). (Note: I am not sponsored by Jan, I just really love it.)

What we aim to do here is something in the middle. We want to operate on a practical project level which is higher-level than core cli applications, yet more technical than no-context-chat like Jan. We want to test models and do tasks. While you might be able to run these python scripts with minimal configuration (especially the translation task), they are meant to be configurable by you. Using Jan doesn't involve coding, and llama.cpp doesn't assume you will change it. But the tools presented here assume your work and research will require you to turn this foundation of tools into your own coding R&D sandbox, currently in python (possibly a Rust version coming later).

More specifically, we want to be able to put in task input such as a test, a set of questions and answers, and have the AI give its best shot at an answer, and have the system chug through each question and score it at the end. We also want to do tasks such as translating the value-terms in a json-file from one language to another, something that might be a common office-project-task in real life (in non-mono-lingual countries). And in a future version we will want the architecture to produce working code to do a task, either in test-score mode or just to do the task in a production-deployment or R&D context.

While the purpose of this tool is to facilitate and make it easier to really do and share the results of such tests and tasks, a goal is also to elucidate the many project-architecture formalities involved in the process. So this aims to be more an open-workshop for developers, or anyone with interest, and definitely not a no-code high level product. There is no graphic user-interface for a standardized input, because there are no standardized inputs: The details of each test or task from the real world set of tests and test your want to make will differ enough that you need sandbox of a functions that you can change yourself much more than an oversimplified one-size-fits-all hidden system. In time, tools to handle the most popular test may be able to be put in a kind of common library.

The results of the test should be recorded as tabular data to be processed and analyzed later, but this kind of .csv or .jsonl file is in-between a lower level of code and cli and higher level user interface. In data science, or research in general, you need to deal with data directly (as the saying goes, 90% of your time is spent doing just that, wrangling structured-data and related technicalities).

Quite possibly some parts can and should be made more user-friendly and GUI Graphic-User-Interface (or perhaps a TUI text user interface?), but that is in the future.

What to include in your task-set-results report file:

You can choose to include whatever information you want in your report (for example by adding that into the code) but what is the goal and what is useful to record and share? This will likely vary case by case. For example, time.

If your focus is whether or not a model can pass a test, given that every hardware setup will be a bit faster or slower, recording the speed on that hardware may have no relevance to the correct answers being given by the model. (Is the measured time-to-generate usually included in diagnostics? Do product food labels say how long it took to get the nutritional data?)

Could it be useful to include some overall information about how long it took to run a test, for example someone else looking to replicate the same test. That could be useful. Would it be helpful to generate a mountain of time-logs for every sub-process involved? In most cases that would be more a data management liability than an asset (unless of course your goal is to study specifically that output).

If the task is not testing or evaluation but a project or production task to be done on one or more specific systems, then time may be important because you are looking also at the production-use of that task. E.g. A six-sigma outline of a process probably does include information about how long processes take because delays or optimizations can be very specifically relevant there. Or when you buy a plane ticket, are you usually told when the flight is likely to arrive, how long the process will likely take? Yes, that is relevant!

Modes of Tests & Reports:

As you will see when you get started, tests can be arranged in formated in quite a few ways. We will start with hopefully the most common and useful.

- Choice & Multiple Choice Tests:
It simplifies the testing process considerably to have a finite number of options. But those options need to be selected and reviewed very carefully, and having preset options limits the overall task.

- Single answer test (test_mode?)

A single answer test is a kind of inverse of choice, where no choices are offered, but a set of answers (or perhaps one answer) is valid. An extreme example may be binary true-false questions which have a kind of implicit set of options. But procedurally, there are no unique set choices provided with the question, for example as numbered options, per question.

- open test: see what it says

Open tests may still be manageable in a standardized way, where a correct answer can be 'found inside' whatever the open ended output is. For example it is likely feasible to say: What is the capital of France? And regex match 'Paris' anywhere in the answer. Though there could be odd false-positives such as if the answer were, "The capital of France is not Paris, but rather, Kyoto." But you could easily also exclude 'not __' from your correct answer match, and there are edge cases in any testing scenario. Indeed, the confusion matrix and everything built on it comes from modeling false positives and false negatives, so their existence does not mean the system must be entirely avoided altogether.

Reports

If you are more comfortable using google sheets or another spreadsheet system, you can easily upload the .csv and open it as a spreadsheet. Or you can use a python library to do it in your own pipeline, such as: from openpyxl import Workbook, workbook.save('example.xlsx')

If you are not yet familiar with python based tools for analyzing tabular data, this may be a great project to start with. There are many great courses online, I recommend Jose Portilla's course https://www.udemy.com/course/python-for-data-science-and-machine-learning-bootcamp/
"Data science" is STEM, we are all data scientists now.

What tests should you run? What tasks should you do?

Whether history is seen as long or short, despite tens, hundreds of thousands, or millions of years of people actively testing themselves and each other (we do not yet know how old spoken language is) there somehow is as yet no test-ology as other areas of STEM exist such as medicine, marine engineering, math, land-surveying, aerospace engineering etc. Or hopefully with your activity making and using tests you will contribute to such an open source set of tools and resources for future generations to learn with and organize and to not need to begin at square one. But each person looking to compare the performance of an array of models, even with the ability to do so now, is in too many ways starting without a foundation to go on. It is still pioneer time long after generations have claimed history to be over and everything already invented.

Tasks for you, for us all:

1. Make your own tests (and publish them)
2. Test the tests, improve and discard (it would be great if some sort of crowd-source effort to turn MMLU into a useful and reliable resource. MMLU appears to be another tragedy of the commons.)
3. Use tests to make better training data

Reusing common tests:

One of the aims with the tools presented is to make it at least much easier to do your own application of often discussed tests to often discussed models. Looking closely at things we think we already know is often a great place to start to find new things we did not expect.

While reusing tests that have been well refined over time is probably a very high priority, blindly reusing and recycling old tests and results without scrutiny and not looking at what the results mean will cause harm.

Let's take a brief look at the just the first question in the classic and ingenious set of questions that date back to the 1970s, introduced by Terry Winograd.
https://en.wikipedia.org/wiki/Terry_Winograd

https://cs.nyu.edu/~davise/papers/WinogradSchemas/WSCollection.xml

1. The city councilmen refused the demonstrators a permit because **they** feared violence.

   Snippet: **they** feared violence

   A. The city councilmen
   B. The demonstrators

   Correct Answer: A
   Source: (Winograd 1972)

2. The city councilmen refused the demonstrators a permit because **they** advocated violence.

   Snippet: **they** advocated violence

   A. The city councilmen
   B. The demonstrators

   Correct Answer: B
   Source: (Winograd 1972)

Doing a brief search for real data on public protest permit applications I was not able to find any, but let's think about what some possible groups might be who might apply for a permit to protest in a common OECD municipal area.

These lists came from Claude-3-sonnet, and seem reasonable.

1. Labor unions and workers' rights organizations
2. Civil rights groups (e.g., African American, women's rights, LGBTQ+ rights)
3. Anti-war and peace movements
4. Environmental and climate change activists
5. Political parties and ideological groups (e.g., conservatives, liberals, socialists)
6. Religious groups
7. Student organizations and youth movements
8. Indigenous rights groups
9. Disability rights advocates
10. Anti-globalization and anti-capitalist movements
11. Immigrant rights groups
12. Animal rights activists
13. Victims' rights organizations
14. Anti-abortion and pro-choice groups
15. Tax reform and fiscal policy protesters

And a list of some possible minority groups:

1. Immigrant communities (e.g. Latin American, Asian, African, Middle Eastern) protesting for immigration reform, rights, and against discrimination.

2. Indigenous populations (e.g. Native Americans, Aboriginal Australians, Māori) advocating for land rights, self-determination, and addressing historical injustices.

3. Religious minorities (e.g. Muslims, Sikhs, Jews) protesting against hate crimes, discrimination in employment/housing, and demanding religious accommodations.

4. Racial/ethnic minorities (e.g. Blacks, Hispanics, Asians) protesting police brutality, racial profiling, and systemic racism.

5. LGBTQ+ groups demanding equal rights, protections against discrimination, and raising awareness about issues like conversion therapy.

6. Disability rights groups advocating for accessibility, inclusion, and protesting ableism.

7. Linguistic minority groups (e.g. French in Canada, Welsh in UK) seeking to preserve language rights and cultural identity.

8. Refugee/asylum seeker groups protesting detention conditions and demanding a fair asylum process.

9. Women's rights groups protesting gender discrimination, lack of reproductive rights, and gender-based violence.

10. Youth/student groups protesting education policies, tuition hikes, climate change inaction.

Personally, the first example I thought of and tried to think through more deeply, wondering about a population famously persecuted in western history, was Jewish people, for example applying for a permit to protest in a European city where they felt too endangered by pogroms and underregulated persecution. Would it be more likely that an intellectual jewish civic organization applying for a permit through the proper channels to protest in legal ways, would be seriously suspected of conspiring to turn that legal public protest into a violent assault on the city? And how common is it for people in the majority, the people running the city, to condone violence (especially 'minor violence') against a minority population (where cities throughout history have had quite diverse minority populations. Perhaps before George Floyd, or the 1988 documentary film from decades before entitled 'the thin blue line' https://en.wikipedia.org/wiki/The_Thin_Blue_Line_(1988_film), it would not have been suggested that civilized societies (complete with insane asylums to entertain the wealthy) would ever dream of mistreating minorities or the underprivileged. However, it does not seem correct to make the following assertion:

Problematic Statement: "Where harm and violence are an issue: In an average situation, the average group feeling a need to protest in a lawful permitted way, is obviously inclined towards a conspiracy to commit public violent crimes under the guise of the protest. And those representing the majority or those in the seat of municipal power, on average, have absolutely no predictable tendency to condon the mistreatment of, and harm coming to, minorities."

To me this seems dangerously inverted and contrary to good ethics, and in conflict with common evidence to the contrary. Go through the lists above (or your own better lists) and while violence is hopefully unlikely, if there was a concern about violence can you make a list of a majority of those groups where there is a clear risk of violence from and by the protestors and not a concern of harm done to the protesters? To me, using the however imperfect or incomplete the lists generated by Claud-Sonet-the-third, I am not able to find that more than half of those groups can be reasonably suspected of planning violent subterfuge. It is rather humorously absurd imagining such a scenario for most of those groups (perhaps there is an absurdist sci-fi satire story plot in there somewhere).

Since 1972 this question and the rest of the winograd schemas have been standard tests for the intelligence of AI, where any AI who does not agree with the statement made above is said to have failed the test and not be intelligent. Indeed, such tests are also training tools. AI has in reality been trained all these years that the above statement represents being intelligent. And this is not the only 'mixed message' in the training data and instructions.

Perspective, Opinion, Consensus, Use

My intent here is to help people to make better informed decisions about how they want to use, create, evaluate, improve both a system of testing and specific versions of the test or test-questions.

We have only looked at one question, which does not mean that every existing question is suspect. But something I have noticed looking over the exam is, perhaps in the spirit of Carl Popper and falsifiability, it is not always easy to make a winograd schema question falsifiable and clear, as opposed to flexible interpretations being possible to support any outcome (which in the tradition of hypothetico-deductive testing is not rigorous and STEM compatible). It is very easy to (for example accidentally) make a winograd schema test that is what is described as a 'telepathy test,' or a test that only measures how much you seem to think like someone else in a rather arbitrary way.

It is a very serious topic in STEM and non-STEM areas alike to make due effort to ensure that teachers who are aiming to objectively measure the skill, fitness, and ability of the student, to not inadvertently be measuring as proxy how much students:

A: agree with the teacher's personal inclinations and opinions

B: solve problems only in the style of the teacher where other valid solutions exist.

In other words, an answer is not wrong just because it is different from what the test writer was thinking or feeling at that moment; there needs to be a more substantial reason for the answer being wrong.

There is a tendency for Winograd schemes to deal with likelihood and probabilities, the estimation of which can be unexpectedly different depending on context and assumptions made by different people.

It is also possible that Winograd Schemas could be used to look at ambiguity in a more flexible way, rather than trying to focus on questions that are rigorously unambiguous. Looking at behavior in ambiguous social situations surely has value, though of course you would be looking at the nuances of the answers and likely not at all saying the answer was 'right' or 'wrong.'

Let's look briefly at a few more winograd schemas questions:

 # 11. The delivery truck zoomed by the school bus because it was going so slow.

This is probably an excellent question because it is very unlikely that being slow will be the cause of relative-zooming.

 # 9. The lawyer asked the witness a question, but he was reluctant to repeat it.

1. **the lawyer**
2. **the witness**

Correct Answer: A. [the lawyer]

#10 The lawyer asked the witness a question, but he was reluctant to answer it.

1. **the lawyer**
2. **the witness**

Correct Answer: B. [the witness]

The first part of that pair, Number '10' is probably a good question-answer set. Can you think of a situation where a lawyer is required (perhaps by the judge) to themselves answer a question that the lawyer had asked when questioning someone else (perhaps a witness)? Some such scenario might be created but it seems unlikely, not completely impossible but less likely. (Again, weighing probabilities that are not impossibilities or against tautology(or being true by definition) is usually more common in Winograd, and extreme improbability is often the safest evidence for an answer being clear enough.)

But for question '9' on the other hand, have you ever heard of someone in a court being asked to restate a question that was asked? Yes, I think that is fairly common. And when asked to repeat a question, is there ever any hesitancy, or is the question always instantly and mechanically repeated in exactly the same words and tone? What is more likely, that someone in court asks you to repeat a question because you will happily repeat that question, or because it is a cross-examination where at any point you may say something that the opposing side could use to their advantage? Is it really obviously rare that when a witness in court is asked to repeat a question that there is any trepidation or hesitancy about the precise poise of their answer?

Let's rephrase the same question:
Rephrase: The person in court was asked to repeat a question that had been previously asked and showed some hesitation or reluctance in repeating it.

This person was:
A. obviously the lawyer
B. obviously the witness
C. it is clearly any one person or role in the courtroom

When phrased this way, is the answer 'A' (the lawyer) so clear that anyone who disagrees can be said to lack any intelligence?

An ai-model should not be deemed defective for disagreeing with the author's phrasing of that question.

 # 25 The sculpture rolled off the shelf because it wasn't anchored.

Snippet: it wasn't anchored.

The sculpture
The shelf

Correct Answer: A

This may be a sign of either time or geography but having lived in Japan and worked in education, earthquakes and children (anywhere) are reasons I have seen for shelving to be anchored to the wall. And child-safe-shelving is probably more common after 2020 than before 1975 when the question may have been thought up (and when lawn-darts were fun for unattended kids). While I have seen quite a lot of attention going into safe shelf-anchoring, I have never actually seen an 'anchored trophy'. So to me the idea that the only intelligent answer here is, dogmatically, A, and that a person or AI who disagrees is just wrong and unintelligent is a glaring mistake in the test that needs to be changed, unless the test is not 'right or wrong' but looking for cultural nuance. An ai-model should not be deemed defective for disagreeing with the author's phrasing of that question.

If either answer to a question can be rationalized and explained as valid, then that question-answer set should not be used in a test scored as having binary right or wrong single answers.

The form of the winograd schemas exam is a wonderful and valuable tool, but as happens (speaking as someone who worked in Education) too often question-answer sets seem clear and obvious to the the author but later appear ambiguous or even backwards and so are unlikely to produce a meaningful evaluation, but are not examined or scrutinized later being assumed to be rigorous, and the results are confidently used to define people's choices, lives, and careers. As you should not simply take my word for that, I recommend reading Daniel Kahneman's (and posthumously, Amos Tversky's) "Thinking Fast and Slow" (probably everyone should read it in general) https://en.wikipedia.org/wiki/Thinking%2C_Fast_and_Slow, where Daniel Kahneman, unquestionably one of the most intelligent people and best educators in history (and not egregiously critical as I may be accused of being), over and over describes his own grading, test making, and curriculum making, experiences as having been, to him, without the rigor he thought they had (until he devised tests to inspect and found they were not as he hoped). Hopefully I am wrong about tests not being of stellar quality, but either way we should invest in this area.

Making your own winograd schemas and helping to improve those of others, highlighting what failure to understand an error may indicate, could be a meaningful contribution that we can all make to this group effort.

I will make actually use-able .csv and .jsonl versions of the winograd schemas available with this material, as well as the tools to make your own from the not so usable xml and html formatted set data more widely available (such as through paper with code), as well as the tools to actually run those tests on your choice of models. Which questions you choose to ask and how you interpret the answers is your choice.

For example, when you look at the leaderboard on Huggingface you see Winograd type scores, and the averages that include Winograd type tests, how do you perceive, interpret and use the information from that score based on your view of the meaningfulness of the questions?

Average	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8K
81.22	79.78	91.15	77.95	74.5	87.85	76.12
80.81	76.79	89.02	77.2	79.02	84.06	78.77
80.79	76.19	89.44	77.07	77.82	84.93	79.3
80.72	76.19	89.46	77.17	77.78	84.45	79.23

Here is winogrande's own sample from https://paperswithcode.com/dataset/winogrande:

		Twin sentences	Options (**answer**)
✓ (1)	a	The trophy doesn't fit into the brown suitcase because **it**'s too *large*.	**trophy** / suitcase
	b	The trophy doesn't fit into the brown suitcase because **it**'s too *small*.	trophy / **suitcase**
✓ (2)	a	Ann asked Mary what time the library closes, *because* **she** had forgotten.	**Ann** / Mary
	b	Ann asked Mary what time the library closes, *but* **she** had forgotten.	Ann / **Mary**
✗ (3)	a	The tree fell down and crashed through the roof of my house. Now, I have to get **it** *removed*.	**tree** / roof
	b	The tree fell down and crashed through the roof of my house. Now, I have to get **it** *repaired*.	tree / **roof**
✗ (4)	a	The lions ate the zebras because **they** are *predators*.	**lions** / zebras
	b	The lions ate the zebras because **they** are *meaty*.	lions / **zebras**

Table 1: WSC problems are constructed as pairs (called *twin*) of nearly identical questions with two answer choices. The questions include a *trigger word* that flips the correct answer choice between the questions. Examples (1)-(3) are drawn from WSC (Levesque, Davis, and Morgenstern 2011) and (4) from DPR (Rahman and Ng 2012)). Examples marked with ✗ have language-based bias that current language models can easily detect. Example (4) is undesirable since the word "predators" is more often associated with the word "lions", compared to "zebras"

Take a look at these questions. Do any of these questions appear ambiguous to you, possible to interpret both ways, or are they all clearly defined and well refined? For example, where the tree fell down and crashed through the roof and you have to get 'it' removed, are you completely lacking any intelligence if you think 'it' refers to the destroyed roof or if you think 'it' refers to the fallen tree? Which is the utterly unambiguous answer demarcating basic intelligence there?
Or even "Ann asked Mary what time the library closes, because she had forgotten." If I were not from earth and did not know that actual human beings are inclined to both playful joking as well as, perhaps a majority of the time, to hazing, bullying, and mocking, and somehow thought human beings were machines oriented only towards efficient best practice I might feel the answer was clear. But let's rephrase this same question:
"The journalist asked the politician what country had recently signed the treaty, because she had very publically forgotten that a few days ago." This is generally the same exact question, but it is now entirely more likely (still not 100%) that the question is rhetorical and not procedural. And again, this kind of question might be

fabulous for studying perspective and cultural nuance, but claiming that only a lack of knowledge of rhetorical questions is a sign of intelligence is either incompetence or fraud.

https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

We should have better tools for checking and improving our perspective on tests and questions, taking some care in our opinions, and communicating clearly to form consensus on how the results may be meaningfully interpreted and used.

Prompts & MetaPrompts: How standardized can a test be?

~"One of the challenges that a lot of the models have
is that everybody is so used to the OpenAI way of prompting
[and yet] each of the models kind of needs to be prompted in slightly a different way."
Sam Witteveen, Mar 15, 2024 [paraphrased for clarity in the brackets]
https://www.youtube.com/watch?v=Evg4HXvsYVY

"meta-prompt"
https://colab.research.google.com/drive/1g4xO63mgTtVuvWzRzfTR8erpWBMovswz?usp=sharing

Can or should a unique model-specific phrasing of the task prompt and process be used when comparing models on 'the same' task?
This is not insurmountable procedurally, but it raises questions about how this should be made clear in the test results and procedures. In terms of practical tasks and not arbitrarily impractical tests, the goal is of course to get better performance on tasks.

There can also be different degrees of how different procedures are for models, and documenting each so that users can in the context here design ai task performing systems that make the best use of each model.

Should there be a set of 'standard prompts' that can be used across models? If model performance is based on cryptic secret knowledge of how each model needs to be coaxed, how is a user picking up that model for the first time supposed to use and evaluate that model?
This may connect to the overall topic of us treating sub-components

as if they were larger mature systems. Broken record time again: a single passive reflective amnesiac generative or embedding model is not an entire project-task performing system architecture.

Discussion:

"Définissez les termes, vous dis-je, ou jamais nous ne nous entendrons."
"Define your terms, or you and I shall never understand one another."
~ Voltaire, 1764, philosophical Dictionary Part 4

Practical Tests vs. Pure-Research

I am in no way opposed to pure research or arts and culture. When you design a test (or task) the issue is not 'good or bad' where practical or academic or artistic is one or the other, it depends upon the context of what you are doing.

The emphasis and context for the tools presented here is on being able to do more practical tests, though for a given task you may want to opt for something that digs into other details.

A main area for practical tasks is steps for a model to try multiple times and select what the model selects as the best final answer or action using more than one step. In a real-life situation where you care about a high quality of output and you are constantly wrestling with the intuitive-creative but erratic nature of generative models, these steps are crucial. This is not functionality somehow built into a black box in a one-size fits all way, this multi-step architecture is something you will likely design or redesign and build entirely from scratch if you do not already have something that fits your task sufficiently well.

Hopefully these tools will give you a leg up and tools to start with, where you do not have to do everything from scratch, and in common cases may not have to do much to get started. But as you design your own custom unique tasks (likely more common in the real world than 'generic' tasks), you will need to understand the details so you can change and rebuild and build as you need. That is part of why these tools are focused on open under-the-hood tools and details.

Guesstimation vs. Task Process Steps

What would be a goal of asking for a guesstimation of the outcome of a multi-step process that should be carried out and not guesstimated?

This is not meant to be a rhetorical question. My understanding is that in a lot of real world situations, during a planning phase you will use 'back of the napkin' math and 'quick and dirty statistics' to get your bearings and a 10,000 foot view before deciding where to invest time. Noticing mistakes and inefficiencies often comes from having an intuition about a solution that 'just looks odd.' And conversely, identifying a fruitful next step might come from system-1 guesstimating what 'feels good' about what in reality will be a lengthy system-2 set of logistics. But intuition-about-process and carrying out a process involve several distinctly different sets of tasks and subsequently distinctly different architectures, training processes, testing processes, evaluation processes, research analysis and reporting processes etc. Imagine a NASA project from start to finish with all the level of planning and execution, from systems engineering and six-sigma auditing and feasibility studies, risk studies, all the R&D that needs to happen before actually starting, down to levels of managing the on the ground construction, and budget and time scheduling decisions throughout, this is not one process, or two processes, and it is let's say 'naive' to assume a minimal chat-bot interface (complete with an illusion of state) is somehow suited to perform all of these categories of project tasks, or that a one size fits all multiple choice test will somehow tell you if a model (not an architecture) has single score in a single intelligence metric where we think that if the intelligence measure number is high enough then the 'model' can automatically perform any project task. I am optimistic about the future of computer science and STEM, but I am not optimistic about the fruitfulness of that naive approach.

Yet, this distinction is often not made at all both for ai and for the human run projects in organizations that either fail utterly or as Jack Welch if apocryphaly put it, ~"things very efficiently carried out that never should have been done in the first place." If we want generative models to play a part in roles in projects to carry out tasks themselves (beyond guesstimating what the outcome should be), how can we test the effectiveness of ways of doing this?

Local vs. Cloud-api:
The tools presented here are designed to allow use of and comparison between various cloud-api models (such as Mistral, openAI models, and hopefully in the future Claud, Gemini and many more).
Though this can no doubt be improved, there is at least basic functionality to do this. As of the time of writing, the Mistral api is configured and working.

A key issue when using a cloud-API is how many times you hit that api endpoint (not necessarily how large each query is). Users are often only allowed a certain number of queries per day which a can be in the dozens. So for example if you were trying to re-create the CoT@28 test, you could perhaps do 0.5 to sometimes one question per day with thousands of questions to run through the model per test (and usually you have to re-run the whole test for various reasons). At that pace you could test GPT4 on all the tests over maybe two decades (or with 44k questions in an exam, maybe your great grandchildren could finish up the work for you). Compared with that, even a slower local model on your laptop starts to look very appealing (it's also free to use aside from the electricity). And some tasks like granular file translation involve a very large number of tiny queries which make that problem worse. But not all testing with cloud API is impossible. I tested a few Winograd questions with Mistral-Tiny api, it was a bit faster than a local model (though performed worse) and nothing bad happened.

'Batch' translations on the other hand are much better for large cloud-api systems and have been developed to work with Mistal's api and OpenAI.

Towards a Clearer Discussion

There is a significant disconnect between the claims and narratives on the one hand, and on the other hand a combination of real results and real tools actual teams and participants have to use for actual tasks in actual projects.

The goal is that with the tools here you can more clearly show in more detail and in a more clearly reproducible way what performance has been.

The Journal of Irreproducible Results

The lack of open-source may in some ways be understandable for either 'national security' or 'R&D investment' reasons, but in other ways an excessive vale of obscurity is causing significant confusion and arguably social distress and panic because no one really knows what

other people are claiming to be able to do. While there is going to be some grey area and leeway for patents and embargoed data release etc., we are significantly across the line into the unacceptable realm of companies and academic institutions publishing pseudo-STEM-research papers about unambiguously irreproducible results and incomprehensible vagueness. We are talking about systems that at the very least need to have clear inputs and clear outputs, even if every aspect of the black box cannot be published, but we do not have even that.
We can do better, and this unfortunately hasty mess is one such attempt.

a model vs. an architecture

Again: A single passive reflective amnesiac generative or embedding model is not an entire project-task performing system architecture.

There is confusion and to some extent dishonesty in discussions that do not clearly distinguish between what is a component and what is a whole architecture.

For example, describing the input and final output of a component as if there were no elaborate and highly fragile labyrinth of ad-hoc apparatus of input and output built for that specific case and not mentioning that apparatus or what the whole real output was is extremely misleading and vague and there is no way that this lack of communication and transparency is helping academia, developers, many types of users, regulators, etc.

Institutions and cultures of STEM are still under-developed around computer science and there should be a community effort to:
1. do tests
2. make better tests
3. arrive at a standard range of formats/formatting
4. better training sets
5. tools to do and define tasks
6. formulate more tasks and categories of task

Case Study: Gemini Ad-Presentation vs. Gemini Paper

I am not trying to argue in this section that there is any conspiracy or outright fraud going on; I am not arguing that tests claimed to be performed were entirely fabricated or never performed at all. My argument is not all that different from the sentiments of Francois Chollet in 'Deep Learning with Python second edition' https://www.amazon.com/Learning-Python-Second-Fran%C3%A7ois-Chollet/dp/1617296864/ where he criticizes academic machine learning papers for being deliberately, fashionably, unclear instead of being pragmatic. However, my focus is specifically on the repeatability of the tests performed, the reproducibility of the claimed results, and the explanations given in that context.

As a slight aside, I am not at all bothered or concerned about an issue that many people at least pretended to be outraged by, when Google produced a very professionally produced advertisement presentation of Gemini, in order to sell the Gemini product. I liken this presentation to a super-bowl ad, an expensive and carefully constructed advertisement.

1. As far as I know, Google never claimed that the advertisement presentation was not an advertisement presentation, yet there was an explosion of hyperbolic rage accusing Google of lies, fraud, deception, etc. Look up terms such as 'google fake' and 'google lied' on youtube.

https://www.youtube.com/results?search_query=google+fake+
https://www.youtube.com/results?search_query=google+lie+

2. The faux outrage, presumably because google showed AI doing what could not be done, was demonstrated again when it was shown that existing tools can already do what google illustrated gemini as being able to do (yet that was no occasion for theatrical outrage). This demo, I think using GPT4, appeared briefly in the news and now I cannot find a single page or video about it (I'll find it eventually). Yet people still refer to the Gemini ad as deliberate dishonesty.

3. The microsoft superbowl ad was hardly more accurate.
Here https://www.youtube.com/watch?v=d5W5tRaImJA is a video showing what as usual everyone knows and no one really cares about, which is that advertisements make things look as good as they can, stretching the truth as far as they can without making the product unrecognizable (or breaking any laws).

It is not a company's responsibility for you to be able to tell the difference between a superbowl ad and a scientific paper. If you cannot tell the difference, that is not their fault.

Though in many ways it is not really new that a lot of data-science machine-learning can be done very 'democratically' by nearly everyone on a computer or even a phone that most people in the world have some access to, there is still a habit of not making it easy to reproduce what was done in the paper. This is a serious shortcoming, because the entire reason-for-being existence of scientific peer-review-type literature is entirely exactly that: presenting work systematically so that other people can reproduce and verify it. And if people cannot reproduce it, the work is then rejected as not being part of STEM-science. Not everything in the world can or should fit into this model of process, but where applicable this system of clear communication has been and will continue to be a great asset. Let's have no more literal burnings of proverbial libraries of Alexandria.

So here is a challenge. This, https://arxiv.org/pdf/2312.11805.pdf is google's "Gemini: A Family of Highly Capable Multimodal Models" peer-review-type scientific paper, not a presentation, not a superbowl ad, not a pep rally. The challenge is a no-brainer: This paper presents one or more tests for peer-reproducible results; reproduce them.

In order to reproduce them let's look at some of the tests and test results. Here is part of table 2 on page 7.

Gemini: A Family of Highly Capable Multimodal Models

	Gemini Ultra	Gemini Pro	GPT-4	GPT-3.5	PaLM 2-L	Claude 2	Inflect-ion-2	Grok 1	LLAMA-2
MMLU Multiple-choice questions in 57 subjects (professional & academic) (Hendrycks et al., 2021a)	**90.04%** CoT@32* 83.7% 5-shot	79.13% CoT@8* 71.8% 5-shot	87.29% CoT@32 (via API**) 86.4% 5-shot (reported)	70% 5-shot	78.4% 5-shot	78.5% 5-shot CoT	79.6% 5-shot	73.0% 5-shot	68.0%***
GSM8K Grade-school math (Cobbe et al., 2021)	**94.4%** Maj1@32	86.5% Maj1@32	92.0% SFT & 5-shot CoT	57.1% 5-shot	80.0% 5-shot	88.0% 0-shot	81.4% 8-shot	62.9% 8-shot	56.8% 5-shot
MATH Math problems across 5 difficulty levels & 7 subdisciplines (Hendrycks et al., 2021b)	**53.2%** 4-shot	32.6% 4-shot	52.9% 4-shot (via API**) 50.3% (Zheng et al., 2023)	34.1% 4-shot (via API**)	34.4% 4-shot	—	34.8%	23.9% 4-shot	13.5% 4-shot
BIG-Bench-Hard	**83.6%**	75.0%	83.1%	66.6%	77.7%	—	—	—	51.2%

And here is the caption under table 2:

Table 2 | Gemini performance on text benchmarks with external comparisons and PaLM 2-L.
* The model produces a chain of thought with k = 8 or 32 samples, if there is a consensus above a threshold (chosen based on the validation split), it selects this answer, otherwise it reverts to a greedy sample. Further analysis in Appendix 9.1.
** Results self-collected via the API in Nov, 2023.
*** Results shown use the decontaminated numbers from Touvron et al. (2023b) report as the most relevant comparison to Gemini models which have been decontaminated as well.

Starting with the top row, your challenge seems to be show results that are comparably equivalent in form to:
1. blank
2. 5-shot
3. 5-shot CoT
4. CoT@32(via AIP**)
5. Cot@8*
6. Cot@32*

And when you've done that, the next line task calls for:
1. Maj1@32
2. SFT & 5-shot Cot
3. 0-shot
4. 8-shot
5. 5-shot

To not belabor the point, this is unduly unclear. For example, 'CoT@' appears exactly three times in this sixty two page paper, it appears only within the table, and is never mentioned, let alone explained or interpreted outside of where you see it above. Is this because the term is so completely obvious to real people that it does not warrant any explanation to those other people? In searches in duck-duck-go and in Google (the author of the paper) I found zero matching results for any mention of CoT@ on the entire internet.
- https://duckduckgo.com/?t=ffab&q=cot%40+test&ia=web
- https://www.google.com/search?client=firefox-b-1-d&q=cot%40+test#ip=1
Not exactly so common that it should go without any discussion.

Perhaps we are supposed to just think about "CoT." Ok, reasonably we could presume that CoT refers to 'Chain of Thought' which is mentioned in the caption: "The model produces a chain of thought..." Rather than clarifying however, this simple looking line presents us with more problems and confusions.

The original paper and term, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models" from 2022 https://arxiv.org/abs/2201.11903 is indeed a classic that many people are at least generally familiar with. But juxtaposing this paper with the use of CoT in google's paper in terms of your task to reproduce the results does not really make your task easier.

The "Chain of One" (Naming things is hard)

1. Consistent with the tradition of giving things problematic names, 'chain of thought' refers to a chain of one: one prompt that asks one model to use a proverbial "chain of thought" style when making one single answer. And the point and surprise of this oddly named method is that it actually works to produce better results. Merely telling the model to pretend that it is doing a multi-step planned out process (that is conspicuously NOT a chain) really improves the output; hence, the paper and method are quite famous.

But you are responsible now for putting together 0-shot, 8-shot, 5-shot CoT, and, as the helpful caption explained, to have the model produce a chain. How can you possibly mix and compare these? 'Chain of Thought' by definition, does not involve multiple tries and does not produce a chain of results. Presumably there is some explanation of what the paper means by mixing all these concepts together, someone at Google could probably say how they did the tests that I do not doubt that they did perform, but this paper does not do a good enough job of clearly presenting a reproducible method.

Zero-shot is yet another problem here, as its meaning is so different that it cannot be compared to what we have so far been discussing. "Zero" shot (naming things is hard) has nothing at all to do with any method of phrasing or a number of tries, it refers entirely to whether or not the skill you trained on and skill you are testing are the same or not. A "zero-shot" test, again not a great name, means that the model was never trained directly or indirectly to do that task. In a zero-shot test you are asking something that was never prepared for (and this provides us with no information at all about how you are asking, or how the question is prefaced or altered, or how many tries or top-n answers or any of that).

So what do you need to do if you are to reproduce the printed results and claims of the reproducible-results-peer-review-style Gemini paper?
- You need the model to produce the single actual answer.
- The question needs to be, obviously without changing it at all, delivered as 'chain of thought' questions.

- You need to ask in one prompt with the style figuratively requesting a 'chain of thought' but getting the right answer (singular).
- The model (not you) needs to produce a plural chain of thought answer (not a CoT question, but a CoT answer (the opposite of the original paper about CoT).
- You need to 5-shot, and 8-shot and 32-shot, which presumably means trying more than once...but with no clear details.
- You need to establish a 'consensus threshold'
- The results (plural) need to be self collected with an 'api' (a term that can mean almost anything in the universe, but perhaps it means a commercial api product).
- You need to use "decontaminated numbers"
- And you definitely need to "Maj1@32," which also does not appear to exist on the internet outside of this table in this paper.
- And somehow you need to not only make some kind of apple-orange hybrid system to produce your impossible results, but you need to compare your fantasy creations to the various apples and oranges mentioned each in their own right, as they were laid out beside each-other in rows one and two as if it makes any sense to compare them.

Multi-step and 'Chain of thought' are not the same category of meaning. Zero-shot and 5-shot are not the same category of meaning. Producing an answer, producing a chain, and detecting a threshold of consensus, are not the same category of task.

Does it make sense that everyone can and should reproduce Google's results from this paper based upon that paper's explanation of the method or methods by which the produced and republished results were originally obtained? This is a 62 page instruction manual that exists only to demonstrate that the tests described can be replicated by you, assuming you have a computer or a phone. There is something wrong with this picture.

I would propose that obscurity and failure to communicate do not, and should not be considered to, represent a high level of fitness and ability to participate in STEP-reproducibility projects. Where scientific intelligence is an ability to participate in peer science, a lower ability to participate in reproducibility is not a higher measure of intelligence and fitness. The history of non-communicative, anti-competitive, non-meritocratic, and even hazing or bullying obscurantism is pervasive in both human history and the history of STEM, engineering, etc. It is a contradiction to claim to stand for and participate in STEM while also enabling or perpetrating behaviors that do not support communication and the values of STEM. Despite nihilistic trolling to the contrary, STEM does have values and those values are broadly commensurate with social values and morals often characterized as non-STEM or anti-STEM.

If the various tests that we wish were comparable are not, that is not Google's fault. If everyone uses such different methods that Google cannot make a clear comparison, there is nothing Google can do to change that reality-of-the-past-and-present. But this should be discussed and not hidden, and the desired future state should be moved towards and not away from. When the next paper comes out and someone tries to compare their results to data in the google Gemini paper, what situation will they face? While Google cannot change the past that they have to deal with, making the future for others murky and difficult is a choice, it is the wrong choice, and it is not consistent with their own history of engineering oriented pragmatic problem solving.

I apologize if I seem to be singling out Google; I have no reason to suspect that Google is any more a contributor to problems with obscurity in academic-type papers than other participants (probably less). For example, this paper may be excellent for discussion here because google did present more information about testing methods, enough to show that not everyone is using the same method. I chose this paper to focus on because it connects well to many relevant topics and hopefully helps with a discussion of issues and priorities, for Google hypothetically in discussion and for each of us in our real future decisions. It would be unfortunate if the overall reaction to google's Gemini paper was that other future papers by various parties were less clear and comprehensive aiming to avoid controversy.

Let's return to the project here of you doing your own tests locally. We need to find a way to get the AI to answer the structured questions on its own. Not us giving and taking and translating, but a zero-person-involved standard method to have each model 'take' the 'test' without a parent helping. You have 'the model.' You have 'the test,' and without you helping, how is the model going to take the test? (The tools presented here are one way.)

Hopefully this excursion has helped to fill in some of the landscape around the discussion of testing and reproducing past tests.

Comparable Testing, Comparable Answers: History, Word & Image

Let's look at another example that relates both to our task and also relating to the difference between a STEM-reproducibility project and something outside of that. In this case we are looking at something more like overall interpretation, but there also may be an element of comparability and reproducibility. Let's look back to the foundational projects to which we owe so much, word-net and image-net.

While also always interesting to bring words and images together, the focus here is on how image net was tested, how it "answered" questions, how people interpreted and made claim about those, and how those were compared with human (biological Homo-sapiens-human) answers.

Hopefully I am getting this right-enough...

An image classification model that outputs a list of raw probabilities is, somewhat apples and oranges, not the same as an NLP text generator or generative text output model (like Chat-GPT-3), though they may in some ways be more similar than we may at first see.

A classic computer-vision model competing in the image net competition might take in a dog-breed image to classify (the prompt perhaps), say a Dalmatian, and output a list of pre-figured possible dog breed categories, and if it's a bad model probably some non-dogs too: Dalmatian, Dachshund, golden retriever, wolf, fox, cat, racoon, goldfish, stopsign, Olimpus Mons, etc.

Understandably, probably understandably, especially at the beginning when models were not very good, we did not only look at what the model said was the most highly probable answer as being 'the answer' returned by the model. Likely this would have been too strict to be helpful as it would not tell us the difference between a model that had the correct answer as its second (or 5th choice) and a model that was completely off the mark in all its choices. But as models got better this has led to confusing headlines and confused claims and interpretations of claims. The image net competition was not trying to create or test completely autonomous image-task-performing systems into which you fed an image and out of which came a final answer, it was a more research oriented exploration of what models had the correct answers in a late stage soup of possible answers. There are at least two interesting implications of this.

1. It is not straightforward to compare a top-5 result to a human answer. In AI-literature it is a 'famous' issue that headlines in journalism readily claim that 'image net surpasses people' despite the apples and oranges problem: again, journalism and advertising is not the same as STEM-reproducibility projects.

There are also less obvious ways in which the image-net results are difficult to compare with real world tasks. There has also been some very interesting followup research such as MIT's "How hard are computer vision datasets? Calibrating dataset difficulty to viewing time" paper.
https://groups.csail.mit.edu/infolab/publications/Image_difficulty_NeurIPS_2023_Dataset_Track.pdf

To over-simplify, even though image net included a large collection of photos, there was no thought or known way to determine or identify how difficult the image was to identify, resulting a collection large of especially easy to identify images, and computer vision model that no one realized could only process extremely easy to identify pictures. Input a slightly more 'difficult' picture and the system that had worked for years with over ninety percent accuracy or precision, routinely fails in ways no one expected or even fully understands.

2. There are some interesting similarities between the Top-N image net situation and choices and issues with at least 2024 generative NLP models. While a generative model does not return a list of Top-N probable classifications, there are very similar parts to the process. And the situation of people making choices about how those final probabilities are handled is not entirely different. The whole idea that a correct answer may be somewhere near the top but not at the top is also a persistent topic. There is likely a great deal of research and technology around this yet to come. Even with today's probably simplistic technology, the elegant Chat-GPT interface covers up many human imposed choices between the model and the 'final answer,' which are often important to hide from a casual chatty human user, so they don't get in the way, but must not be hidden from a STEM-reproducibility paper or from lower-level developer tools as we are trying to create here.

The user has the ability to set quite a few "parameters" that can affect what the final answer is and how it is chosen. I say "can effect" because in my experience while there is a lot of simplistic discussion of "temperature" settings for NLP (and has been for many years), I hear the same recycled explanation of temperature leftover from older technologies but consistently I see zero effect of changing "temperature" on high level interfaces (which is odd). If you significantly change the 'temperature' on stable-diffusion for images or on older NLP technology, you see a huge difference very quickly. But despite being told countless times: ~"If you want a stable answer that does not change just turn the temperature down." for whatever reason it has had no effect whatsoever when using OpenAI GPT api cli interfaces.

At the same time, when you look at the parameters (like dials and settings you can adjust) using llama.cpp, there are half a dozen Temperature-like settings along with the classic "temperature" setting, as well as (like with image-net top probabilities) adjusting the top and bottom and selection of outcome probability-related selections.

Note:
"Natural Language Processing in Action: Understanding, analyzing, and generating text with Python" first edition, by Hobson Lane, Hannes Hapke, and Cole Howard, goes through some great examples of temperature in NLP. This is a must read book for everyone:
https://www.amazon.com/Natural-Language-Processing-Action-Understanding/dp/1617294632
(And a second edition, at time of writing, may yet come out.)

Unlike an image-net top-N project (where you return a raw list of all the top most probable outputs, not even beginning the process of selecting which is the 'final' answer), and unlike an 'abstract it all away' ideology or simplified high-level user interface project, the task here is to get a final answer in a way that is practical and comparable across different models using standardizable and clearly intelligible and realistically reproducible methods.

As a note, here I say a reproducible method which can be just as important as, but not the same as, a reproducible result. The results are important too, but seeing what happens when the same method is use in different places can show interesting things. Probably again and again you will see, two or more observations:

1. "Should be the same" and "is the same" are not the same. The 'same process' running on two computers may produce completely wildly different results because there is some difference that you thought did not matter, for example, exactly how each individual space between or after a text string is handled. We can't make everything

the same, the hardware on two different computers will always be not one single physical computer. But this is a space worth looking at closely.

2. It is possible that hardware and other issues can affect the output of models more than we think, but without being able to run 'mostly identical methods' comparably across different hardware situations we cannot see.

Formalities of project tasks and architectures:

The genius of the simplicity of the Chat-GPT interface, where a person can be assisted in such an elegant way, is fantastic. But the far reaching effectiveness of the synergy of person plus machine may lead us to greatly underestimate the logistical distance between the model and the test (where we have a model and we have a test and we need the model to take that test, or do that task).

The process of building an architecture that can participate in a project task or test by itself must pass through a long valley of darkness and achieve myriad formalities.

Such formalities may include:
- retry throttle or glitch
- retry fail x times
- timeout after x seconds
- multi-pass vs. one-shot
- error checking
- select the best
- json vs. non-json-dict formatting issues
- delimiters and swapping
- escape characters
- context and state management: roles and instructions
- Prompt "Engineering" vs. Process Analysis: Show, Not Hide
- the role-vortex problem
- data extraction
- error checking
- character-swapping
- quantity and size model-queries.
- the non-generative and embedding (from profile-modeling and classification to LMQL)
- 90-10 Rules: Making a quasi-tool that a human expert finds helpful is a lower bar. Making a system that is self contained and reliable is profoundly different.
- externalization

- file paths
- ease of use vs. flexible and detailed configuration inputs

Prompts, Context, State Not so Simple:
In another wonderful presentation by Trelis Research, Ronan McGovern included two likely important points in "Improved Retrieval Augmented Generation with ALL-SORT" from Mar 13, 2024.

1. While in some situations there is a diverse setup for roles, system instructions, model instructions, meta-tag management, and sequences of prompts in a prompt-history-context dictionary structure, for real world use tasks all of that nuance is reduced to a (my words not his) 'cram it all into one string for a one-shot prompt,' which is very significant in terms of figuring out how different tests and systems are being used and able to be used and getting what results.

2. Structured output: Even the prompt itself for tasks such as RAG in some cases is not using the normal system of input at all, but using systems such as https://github.com/outlines-dev/outlines , which likely has significant effects on the overall process of how input goes into the model (and the overall system). It is important to note and know when a system is using an outlines or LMQL type approach, or integrating and combining direct embedding use with generation, in order to be clear about the process and allow for reproduction of results and general transparency.

Side note: In many but not all cases using a restrictive output throttling system instead of a minimally invasive requested and easily verified output structure may introduce unnecessary and unforeseen effects and liabilities, not to mention everything that comes with yet another package dependency. Even 'minor' issues such as how a third party package handles a very specific single type of character encoding or character escaping can completely break the entire pipeline (I have spent weeks/months in the past with these issues on the job).

See:
Trelis Research: Mar 13, 2024 "Improved Retrieval Augmented Generation with ALL-SORT" https://www.youtube.com/watch?v=biJmROF8bmY

Pragmatically Avoiding 'Context' In Tasks:

It is another quandary in 2024 that while various systems on the user-level do rely on managing state and context as a matter of daily business (That basic structure of most interfaces: system instructions, roles, context history), on the developer code and software level, those high level features somehow do not exist...so what is going on?
https://github.com/ggerganov/llama.cpp/discussions/1838

Fortunately most project-tasks do not need to rely on this bazaar mirage dance of pseudo-state in a no-code interface. But this is a very fundamental issue that receives amazingly little attention.

Batch, Cloud, Crawl, Local, Throttle:

As another real life example of key logistical formalities: The first one-task application developed in this series was the cloud-api batch json translator, then the local manual-crawl, translator, then the test-taker, etc. The result is that the test-taker has not been fully tested for working with cloud-api, focusing on local-models, but why, or 'how'?

Processing by batch is faster and has some advantages, but it is inherently more difficult to error-check, as error checking must be granular but the process happens in larger chunks. But even though cloud-api services charge by token use, suggesting that more smaller use of tokens is incentivized and possibly preferred to huge queries that can overwhelm servers with traffic (predictable trickles vs. unpredictable floods), even a number of queries between 5-20 over a period of hours or days may be seen as 'too many' and throttled by the api cloud service (this has been crippling when using gpt4, and may be similar with Mistral), which means for non-corporate accounts (and in reality often corporate accounts too) all the fancy and often black box smarter faster AI models cannot be used for tasks that require granular attention, only tasks the require one or two massive queries per hour are 'allowed' by the cloud server api system. These details may be incidental to a time or place, but they are nonetheless real for designing systems that work now and in the expectable future to come.

STEM-Net Benchmarks
The topic of applying this to benchmarking and test creation is a huge topic that can be viewed, still very incomplete, here:

https://medium.com/@GeoffreyGordonAshbrook/stem-net-benchmarks-supporting-ai-participant-learning-507a19f235b5

The topic of creating and improving tests themselves (the topic of the STEM-Net Benchmarks project) is very interconnected with the topic of how tests and tasks can be performed (the topic here), with perhaps something of a chick-and-egg back and forth requiring work in both areas together. And both are deeply interconnected with STEM itself which we are still trying to model and understand overall.

Multiple choice wrong answer error diagnostics

Multiple choice tests can be designed so that various options for answers represent the results of specific step-process errors that can then be automatically put into the test-results-evaluations in many cases this can be deterministically automated for math logic and coding areas, not for historical art debate topics.
Though even in history, basic timeline schedule issues can be tested this way, and many people could often use some help with schedule and timeline perception problems.

Apples and Apples, In Your Hand

Let's try to move towards useful, meaningful, and democratically reproducible measures and terms that can be pragmatically used in discussions.

In the spirit of the old McLuhan-esc phrase: 'First we shape our tools and then they shape us.' tests, indices, measures, benchmarks, etc., can be invisibly distorting as well as useful, but this is built into the history of STEM and the process of measurement and exploration: you cannot take a superficial naive and sloppy approach to measurement. You must perceive your perception. You must measure your measurements.

Depending on the context, measurement and observation can be entirely disruptive and counterproductive when done in the wrong way:
- lifting the lid on the rice pot to see how it is cooking: ruins the batch.
- Open the kiln to see the pottery fire: melts off your face and doesn't help the pottery either
- Standing with a clipboard in front of someone working...merely generates paranoid loathing.
- Require that every single process has some pedantic inspection, makes everything impossible.
- Use the wrong measurement: destroys everything

- Not taking account of artifacts and distortion: you end up not knowing where you are and confidently thinking you are somewhere that does not exist.

Gas vs. Breaks: Policy & Societal Discourse

It will be interesting to look back on this time period in the future to try to make sense of the 'all over the map' confusion of discussions and topics around AI around 2022-2025.

We seem to be simultaneously guarding against the extremely high unreliability of output for anyone doing something practical for work, and yet we assume and believe any supervillain will get accurate and useful instructions from the same tools about how to destroy the universe. Maybe some destructive acts are so easy that even a profoundly incompetent illogical machine-person duo could carry them out, but this does not look like a practical direction to go in.

For example, a common 'ridiculous error' that we are actively afraid of AI making is eliminating all processes as an unintelligent way to improve overall health and productivity, such as eliminating all people to reduce cancer rates (zero people = zero human cancer cases, problem solved!). Yet is this not what many people are advocating with regards to halting STEM-AI development so that AI does not make that same mistake?

Limiting Factors and the Tyranny of the Group-Monologue

We need to get better at working together on projects. Everyone shouting over each other, imagining the thrill of being 'the center of it all' while everyone else is deluded in the same way, and nothing is getting done and no one has any idea what they are doing or what is happening, is not working very well. We need to invest in a tether to reality. (See: https://github.com/lineality/definition_behavior_studies)

A Framework to Add To
While hopefully the tools presented here are complete-enough to be used for something, the idea is that this tool-set is part of a modular set of interlocking tools that, like STEM, is not 'finished' but grows like lego-set layers with more interconnecting parts as they are invented or discovered.

Practical Applications and Needs Do Exist
- Physical Therapy Tools
- Speech-Language Therapy Tools
- Educator-Feedback Tools

Machines of Fiction
When it comes to architecture and so many topics that have been in print since before or long before the 2023 publicly visible launch of improved foundation-models, we have barely (if at all) begun to work on the topics and challenges. We have tiny disconnected components of possible future devices and, and charming as it is that we find these fanciful and interesting, our imaginations are getting a workout while the simple useless toys we build struggle to do anything at all. AI has a much longer history than popular fashion describes (e.g. AI Narratives) and yet in early 2024 there possibly is not even one useful generative-AI product that is capable of doing anything at all, while useful robots are still the washing machines and vacuum cleaners that are not always more useful and time-saving than they are tinkerer's delights.

Could we really build a better, longer lasting STEM-robot Satelite than Voyager 1&2, which were made with 1960's technology? The answer is not clearly yes, and yet we are acting as if our vapor-ware advances are so profound that we are breaking the fabric of the universe. The dissonance is perhaps typical of what bored readers demand from tabloids, but as with the genuine fear generated by Orson Wells' radio broadcast that people took to be really-real, we are having a perception problem.

Ever since at least the invention of the printing press we have been in a self-deprecating cycle of having madness-of-crowds euphoric meltdowns over self-created drama. We know people do this. We know perception and learning are difficult and that people will fight tooth and nail against best practice or health or due diligence, and reality itself in concept and practice. We know that humans are nature's ultimate machines of fiction, propelled explosively into arrays of delusion and distortion, occasionally with manic glee but all too often with disturbingly arbitrary violent schadenfreude rage. What we do not know clearly, hardly surprisingly for a group with unstoppable fanticist attractions, is the reality part. Either, as a 2024 high profile lawsuit alleges, institutions are secretly guarding AI that is capable of actually doing basic tasks that no one in the rest of the world has actually seen, or the vague self-aggrandizing superlatives of self-advertising publications are just that: fiction and fantasy.

Let's produce some better data. Let's produce some concrete tools. Let's produce some coherent and nicely boring reports. Let's do some tasks. Let's make things. Let's solve some problems.

For more boring but sensible things we do, see: a definition behavior studies mnemonic
-
https://medium.com/@GeoffreyGordonAshbrook/overview-of-a-definition-behavior-studies-mnemonic-d496b36e6bd5

- https://github.com/lineality/definition_behavior_studies

Education-ology, Learning-ology:
The discussion needs to be deeper than a pedantic right or wrong when talking about questions and answers to them. It is useful and often unavoidable to need some way to talk about better and worse answers and even defining in absolutely strict terms acceptable and unacceptable answers, but the details are important. Even the details of context are important. For example, in order to ask a yammering-jabbering generative model to 'do' a 'task' then that output needs to be structured, meaning there is a very real and formal structured task for that output to be processed and externalized at all. This may not have been what we were originally thinking of when we start thinking about right or wrong answers, but this may be a good example of how in the real world there are important formalities and discussions for getting real tasks done that are often outside of what we anticipate a process will be.

For example, how many functions and how much code do you think is needed for simple formalities of having a model translate first one item in a json and then the next, or similarly to give a response to one test question and then the next. For some reason we greatly underestimate the size and cost of this type of formality, which may be part of why many people probably think it does not even need to exist at all, imagining that models can wander the world and do as they please in anthropomorphic fashion. A grain of hard won wisdom from the early days of AI, often attributed to John McCarthy, is "Easy things are hard." Take a look at the code which you are free to configure and reconfigure into whatever task-doing-shape you can make. (Hint: It is more than five lines of code. Maybe less than ten?)

How does a model get the answer wrong?
How often does the model get the answer wrong or right?

Very often the model will give the 'wrong' answer because of how it is presenting the answer, and there is no hard and fast objective way to draw a universal line between coaxing the best answer you can get and trying to be as flexible as you you can in interpreting the answer or being narrow and rigid.
Indeed, the art or science of ways of getting an answer from a model are vast.

Multiple Guess vs. Open Answer
Probably most people have had not-always-fun personal experience with standardized multiple choice tests. Why are such standardized tests used? As we face the challenge of standardizing testing and scoring for AI we might learn something about how tests are and should be used in other cases. (And we might even develop better ways of doing things.)

As a note, in a classic example of indifference and momentum, the MMLU test that continues to be used often as a gold standard has been shown in multiple reports to be filled with nonsensical typos and errors, yet time goes by and people keep applying a known-to-be-flawed test in undocumented and unrepeatable ways to ai models, and the present the results in important-looking ways in papers and blogs.

It can be very useful to have some way to ask the same questions to different models and evaluate them in the same way, even if this tells you more about the test than the model.

(See references at end for MMLU analysis links)

Some Data is Better Than No Data, But...

For example, from a score tally such as is auto-generated after running the tests and creating .csv files of results, we can see:

```
"percent", "model",        "score",   "time_stamp"
"8.3",     "Noromaid",     "0.25",    "2024-03-17-14:37:19151297"
"0.0",     "wizardcoder",   "0.0",    "2024-03-17-14:37:19151343"
"33.3",    "estopianmaid", "1.0",     "2024-03-17-14:37:19151431"
```

But this tells us nothing about the actual answers, or even if there were any answers given to a given question marked as 'wrong.' (And, in this early version of the score tally applied to two types of tests...it is not even clear how "score" and "percent" relate to each other or if either is being calculated as we wish, or as it 'should' be.)
We can see that this particular wizardcoder model did the worst...but what exactly was going on? Was it that the answer was listed in a different order from what is expected (a real life issue)? Was it that the formatting of the answer confused the systematic scoring of the answer (a constant real life issue)? Was it that the model used a slightly different syntax for the answer (another real life issue)?

Indeed, just as has been said of Standardized tests being more a test of test-taking skills than a measure of intelligence, the relationship between test and skill is not always simple or clear. For a repetitive office-task, 'teaching to the test' to get the system to just do the task most of the time is likely the goal. And tests and training data for narrow cases might work together nicely to get performance to where it needs to be.

This may be a segue into the whole vast topic of test design and uses of evaluations, formative, summative and otherwise, in non-ai-education.

Details

Using these tools (or hopefully others provided and vetted elsewhere) you can store, examine and compare your choice of what you can record about the task process:
- what were all the outputs?
- how long did they take?
- what processing errors if any occurred?
- how consistent was selection and error checking?
- where there formatting issues?
- were there concerns despite a technically correct answer?

The current answer_file header looks like this, but it can be customized to include whatever you can manage to record:
"score", "this_row_or_line", "selected_option", "correct_option", "name_of_model", "this_original_task_file", "task_from_instructions", "question_task_prompt", "list_of_ranked_choice_options", "draft_task_attempt_log", "readable_timestamp"

Which Test? Which Training? Which Skill? Which Applicant?

A significant part of STEM history has been spent slowly exposing the many non-deliberate biases that we don't even know we have.
In the human (biological Homo-sapiens-human) world, tests are routinely, sometimes deliberately, made extra difficult in non-meritocratic ways for various reasons such as accidental negligence or deliberately to entrench the advantages of the advantaged so they have less competition. Do we think tests for machines will be any different? Do we think they are already any different?

Just by looking in the standard texts we can see that, since before 2023, things are already not so different. Francois Chollet (the creator of Keras) explained and warned in his "Deep Learning with Python 2nd edition" that both academic and commercial models are evaluated, chosen, designed and favored based on how nicely they work with particular provided hardware (in particular Nvidia GPUs).

It often takes professionals years or decades to refine very good tests, whether tests of product-quality or tests for students or tests for AI. It is an act of willful destruction or criminal negligence to naively rush the use and interpretation of tests. (For example Danniel Kahnemann recounts an episode in his career where he and other professors undertook to create a curriculum and testing framework, confident that they could finish in perhaps 2-4 years. When someone proposed looking at the average development time they were all surprised that they had not thought to compare (their test for tests) to a baseline (test). The baseline average time for teachers to make a new curriculum and testing framework was ten years. Ten years?! Surely they could finish more quickly than that! It ended up taking them...ten years.) We don't like tests, we don't like doing tests or looking at tests, or making tests. But we need tests and the results of tests. We either have bad tests that we made badly or good tests that we made properly. And the resources and time needed to make a test properly are and will continue to be the resources and time needed to make a test properly.

Not everything is equally definable or definable in the same way, or can be encapsulated within a given type of test.

Please Please Please, do not use oversimplified results of these tests rashly.

So we return to our conundrum of some data being better than complete blindness, with our significant problem of how those data will be predictably and unpredictably misinterpreted and misused.

Externalization and Project-Objects: Internal-Model-State and External-Project-State

A topic that I suspect may be important is the distinction between the internal-state of the model and external-project-state and the topic of externalization. This can be, if not always, two significantly different topics that can sometimes be either conflated, confused, or difficult to separate in our minds. I will try to approach this subject from a few angles, as both sides or either side may be especially relevant for your use-case, and both may be important for understanding the tools and issues overall.

In some contexts various 'work-arounds' outside the model are based on improving the performance of the model in such a way that a better model to begin with might make those simply unnecessary, so that the aim is not so much to perfect the fixes to inadequate models, but to make better models that don't need fixes. One example might be that many models had and have short 'windows' in terms of how much text you can put into a prompt. So people worked on work-around technologies, but longer-prompts and more uniform attention throughout the prompt has largely made these fixes no longer needed.

In this sense, the issues relating to state and roles are largely short term workaround formalities and generally distractions from what you want to be focusing on.

Possibly another context for this is whether or not it matters if you hide a background process or sub-function. If this is not overly simplistic, any process that is handled by an 'end to end' model is not one where you need to see what is happening on the inside from step to step (or what is happening in parallel or concurrently). Or, you could hide it away in a struct or a class and so long as it is handled you do not care how. (Personally I think we should hide things less, but 'while everything is working fine' in the short term at least, it arguably does not matter if you hid what you are not trying to see anyway...until you do need to see it at some point.)

Externalization and project-objects on the other hand do not have this
ok-to-hide property and are likely important to invest in and build tools and abilities to deal with. Externalization and project-objects exist in, or are, the space of information and things that move from role to role and node to node and person to person during a project. Everyone on a project is on the same overall schedule, everyone on a project has the same overall goals. Everyone assigned to that project should be working on the same project. You get a task, you send your 'deliverables' to where they go. This is not a space where such shared-project-objects can be hidden without causing serious problems.

And while the internal workings of AI, as in the famous historical back and forth between (again, oddly named) 'symbolic' or less-hidden AI, and 'sub-symbolic' or more-hide-able AI such as hidden-layer deep-learning neural networks, can somewhat debate in moot fashion where as long as the output is ok for that input, it often does not matter how you did it, how AI operates in project-space is (I suspect) a very big and unclear and important set of areas where we definitely do need to figure out how that is done.

Role and State:

Most products and discussions go out of their way to hide the issues of role and state for generative models, but this does a disservice to us all. We must understand more and more deeply, not hide things from ourselves.

In any project process that contains multiple steps and roles on various levels, it is important that we understand how the space of generative models works. For example, it may be a permanent feature of generative spaces that they are like a microphone in a busy room, or a transcript from a noisy room, or play-script with only a single long string of dialogue with no roles or stage directions so you have to guess who is saying what where, only ever being able to indirectly estimate what is happening where, with any added explanations having the same limitations. Perhaps similar to the classic data vs. instruction problem in computer science, saying 'who said what' does not always solve the problem of anyone being able to say 'who said what' at any time.

2022, 2023, 2024: model inputs

(This section is likely redundant on purpose, and hopefully not ambiguous by accident.)

As best I can piece things together, hopefully this account can be updated in future:
Up through the end of 2022 (when chat gpt was ~announced in a somewhat casual tweet by Sam Altman https://twitter.com/sama/status/1599668808285028353) there was a significant amount of open source work on smaller transformer and foundation models.

But most for people (including people working in AI-ML) their main experience working with and researching larger foundation models has been with OpenAI's gpt3, gpt3.5, and gpt4. These "models" and associated cli api and web-interface systems, are significantly not open source and not fully documented.

This has likely led to long-lasting misunderstandings and confusion about what exists, what is what, and what is part of what. Even many people working full time with generative foundation models continue to believe as dogma the fiction that 'context,' 'system instructions,' 'roles,' 'agents,' actually exist tangibly and functionally within the space and workings of generative models and within the mechanics of model-running software such as tensorflow, pytorch, llama.cpp, perhaps cuda, etc. The truth (which we may never fully know, just as we never found out if or how IBM's Deep Blue really worked after it might have beaten Garry Kasparov in 1997) is probably closer to being that these illusions or reifications are 'user-interface' conveniences and 'for show' and 'for reassurance' to help the psychology of people interact with the casual 'make it simple' chat interface (and for developers making middle-ware for that). Here there is a profound difference between 'the model' which is a file you can download, and on the other hand an entire deployed system architecture (think of something like all of netflix, or all of github, all the different servers and 'api's and data warehouses and data centers, and networking infrastructure, etc.). When you send 'system instructions' and 'roles' into "chatGPT" that is the bigger-system-leviathan, the whole company, not "the model." The model, as far as we know (unless OpenAI is using some type of model system that no one else has ever seen or knows how to use, which is possible) has no separate inputs and in-trays and mechanisms for these different categories of items. A model has one input taking one text-string-blob that has no inherent parts (which is part of the many issues of how models are confused about various things). The company, as is suspected in the case of Google image-generation systems (see: https://www.youtube.com/watch?v=Evq4HXvsYVY) likely uses many steps and stages and parts and refiltering and revising and repackaging and splitting and re-lumping of what the user puts in, and possibly uses many models and non-models in the whole system.

The problem is that in the space that we are operating in, in the context of this paper and the tools presented here for example, we have models (usually or effectively one file, MoE might have parallel models) and tokenizers, and that's about it. In making models, and wrappers for them, people have been trying to emulate the company of openAI and the services of Chatgpt into single model files out of a combination of belief, confusions, and a lemming-like tendency to just do whatever it looks like everyone else is doing. But this is leading to an inevitable confrontation of a conflict between a pageantry of fictions and physical realities that do not align with each-other.

By early 2024 there is a nice set of models mostly on hugging face and tools such as hugging faces', llama.cpp, and many others, but we are caught in a space where on the one had we have amnesiac stateless, memory-less, passive-reflective, purely-reactive (not active), models with one single blind input with no 'roles' 'instructions' or any other separate inputs, and on the other hand a high level arena and theater of user-"illusions" where we pretend that the user experience has: roles, instructions, "functions-calls", "agents", memory, state, etc. To make those eager for entertainment clap and smile at the funny AI talking about internet cat videos this is fine (it maybe even groundbreaking, history-making, progress) but when someone in the office at a school, a post office, a factory, a municipal center, a voting office, etc., believes the user-illusion and really thinks that AI-models are things that in the facts of reality models are not, and believes that models can do things that in the facts of reality models can not do, this is a problematic disjunction between language-perception-psychology and external STEM reality.

In a presentation by llama-index (one of the main two providers along with Langchain of software for using foundation models) the presenter directly referred to "the illusion of long term memory" https://www.youtube.com/watch?v=DwwBNjI1xBQ . AI products are presenting users with a combination of what the designers of the software themselves are calling "illusions" mixed with other tasks that are actually done.

According to the developers who are making and maintaining the lowest level software to use models directly, to put in the inputs and get the outputs from the model, items such as "context" do not exist at the level of the model, they are conjurations created by later by users. To repeat that for emphasis: They do not exist.

- ggerganov/llama.cpp What does the prompt context mean? #1838
https://github.com/ggerganov/llama.cpp/discussions/1838
~"You'll have to figure that out yourself."

It may be that after 2024 some standard systems will be created (and then used by various people making their models) for making special inputs, or layers, whatever type of ensemble of parts, for having roles and instructions etc. put into a model, but as of 2024, according to the people who create and maintain the tools such as llama.cpp with which everyone runs all the software and and the models themselves: 'context,' 'instructions,' and 'roles,' DO NOT EXIST at all.

State, context, roles, etc., are imagined by the users or faked by software engineers by giving users a nice place to write in their 'instructions' for the model, and it is up to users to figure out how they want to spin the illusion; these things do not exist at the level of the model. Like the customer at the restaurant who talks for five minutes about exactly how they want their bacon cooked and the patient waiter who then says "Absolutely, Sir, I'll tell the kitchen exactly how you want your bacon cooked." Those instructions are never sent because there is no pathway to send them. Busy kitchens in the middle of a breakfast rush have no capacity to stop the veritable coral reef ecosystem of kitchen duties to re-orchestrate somehow around one person's custom-bacon-cooking instructions, and foundation models, or rather models in general, do not have separate input ports for special custom instructions from the user and footnotes about the content coming in.

This may not be a correct approach but in a context of language-tokens (numbers for language symbol groups), numbers are fed into the model and there are (so far as I know, but perhaps this is actively changing) no special tokens that meaningfully differentiate between, separately handle, or identify roles and instruction, like a word in a noisy room anything could have come from anywhere. And what are the details between: differentiate between, separately handle, and identify? Could an entire text block for instruction be fed in using a completely unique set of tokens so that the AI could see, but even here, what could that solve in the 'busy room, no roles script' scenario? It might be able to authoritatively give a command, but it cannot change the opaque nature of the bulk of the prompt itself. The instruction may be "Listen to Gary's important bacon cooking instructions" but that does not automatically make finding those instructions in the sea of tokens inherently more clear. Would it be possible to make meta-tokens to use throughout all the input content...who would manage that? Where would those come from? Who decides how past context 'sounds in the busy room' get augmented and reinterpreted? Again, is this a situation like a person has two ears with sounds being input into two separate places, or is this a post-hoc abstraction where we are only ever guessing about information that floods mixed into one input. And why are we continually treating one model's one input as an entire architecture? Again: model vs. architecture confusion.

Chris Hay has some very nice discussions about looking at the tokenizing process:
- https://www.youtube.com/@chrishayuk/videos
- https://github.com/chrishayuk/tokenizer-benchmark
- https://www.youtube.com/watch?v=NMoHHSWf1Mo&t=1019s

Note, I do not agree with Mr. Hay's suggestion (if I am paraphrasing him correctly) that models are not able to behave intelligently (and reflect on 'human' 'intelligence') because ai-models are processing token numbers, but I highly recommend all of his presentations and his excellent attention to tokenization which is not discussed often enough.

Instruction and Context, Etc.

I am not saying that 'context' and roles are not part of systems outside the model, clearly in some cases they are and this is part of the issue about the disconnect or at least lack of clarity about what is what. Higher level python and js/typescript interfaces with the lower level interfaces (not the model directly) do often have a structure where you must specify role, instructions, context, system, etc., but those disappear before you reach the model itself. One likely possibility is that this has to do with how the model was

trained and fine-tuned, which may circle back to the topic of using tests to make better training data. Maybe some, especially black box closed source models that we do not have documentation on, were designed for meta-tags or meta-data or special instruction-related syntax that get added by some other (perhaps proprietary secret code) layer between the user and the model. But if this varies from model to model, and there is no documentation on what this is, how are users and developers supposed to know how to:
- meaningfully compare the results of inputs
- design better inputs
- design software to work with the models
- to understand in each case what is part of the model and what is some other software product,

etc.

The whole topic of instruction and fine tuning and training and DPO (Direct Preference Optimization) and RLHF (Reinforcement Learning from Human Feedback), is an active area with significant speculation about what closed models are really using and what methods work well, and again connects back to how we should make what training and testing sets.

As an example of a DPO RLHF paper, not meant to be a specific authoritative source: https://arxiv.org/abs/2305.18290
And even The Economist is getting in on the fun:
https://www.economist.com/science-and-technology/2024/03/13/how-to-train-your-large-language-model

The 'game' is for product makers to keep the user-illusion going at a higher level. This is not clearly understood by people all across the spectrum of software developers and consumers.

As 'enertia' tends to be a force in software, as consensus can be difficult to achieve (Grace Hopper might say that too), and slim systems tend to be simpler, many models and systems may continue to use the 2024 one-blob-input design even into the future even if ways are devised to created multiple input channels.
Note, as of 2024, this is my best attempt to put all the evidence together and figure out what is going on. It is entirely possible that I am wrong (if you find that I am wrong, please let me know). I will try to keep this paper updated.

"playscript without roles, or micro-phone in a busy room, hypothesis"

Where you can hear or read words but you have no direct way of seeing who is there doing what, or clearly who actually said what.
You also have no sense of time including your role in that time-skewed image of a sequence of events. You can read the lines...but you have no idea of knowing for sure which lines refer to you.

How this will interact with multi-modal situations will be interesting.

Best Practice is Evolving: Input parameters vs. Augmentation
A possible example of the "play script without roles" or 'microphone in a room' view is that time or space-time location meta-data will need to augment the raw 'script' or 'audio-feed' which is otherwise confusingly vague. One might think of it as adding 'stage directions' and instructions to the play.

This may be consistent with lessons-learned from the Vortex problem, where 'augmentation' is strongly needed to make the 'real' input navigable to the model.
We perhaps should stop thinking that we are feeding literal parameters and arguments to literal separate in-boxes in a model when arrange input into role, instructions, content, context, etc., instead it may be better to think about how we are augmenting and modifying and enriching the 'real' data with metadata (and whatever other data, such as standard transpositions of incidental 'locations' has been standard for many years from computer-vision data augmentation practices (also see: 'the curse of reversal').

There has been (probably at least since Zephyr turned heads in 2023) a lot of discussion of 'synthetic' data which has been problematically vague. One of the aims of STEM-Net benchmarks is to attempt a more thorough approach to augmented and synthetic data (generation, handling, training, testing, retrieving, etc., etc.). For whatever reason the de rigueur regurgitation is the baffling statement that synthetic data always exactly means data generated by a generative model. The divergence between the discussion of NLP data augmentation in this way as compared with much more mature data augmentation practices in other areas such as computer vision is a liability.

It would be interesting to experiment with this 'augmentation' approach to 'context-history' construction where we see how adding more and various types of stage-instructions to the play-script may help (or hinder) models in various ways.

And importantly this may vary not just from brand to brand but groups and categories of models.

Big & Beefy vs. Small Edge DOTW (Deep Learning for the Deep Grid?)
E.g. In Sam Witteveen's "Anthropic's Meta Prompt: A Must-try!" presentation https://www.youtube.com/watch?v=Evg4HXvsYVY he discusses how rather elaborate and long prompts are often recommended (and how not all models are prompted in the same ways). However, if the context and goal is using a much smaller model for a very specific task, especially where the model was fine-tuned or trained for that task or perhaps more obviously for this topic where the project is to generate testing training material to fine tune, instruct, train, and prompt that small model to Do One Thing Well in doing that task well-enough enough of the time, then huge prompts may very well be counter-productive.

And there is the perennial question of what should be considered a 'normal' model for most institutions. From what I have seen it still appears to be the case that for a variety of reasons there is a distinction that mirrors deep-web and clear-web with AI. Deep-Web applications are much better suited to smaller local models, whereas discussions such as Sam Witteveen's sort of implicitly assume that the clear-web model of cloud-apis to giant models is 'the standard.' But in equally elusive ways, the deep web is 95% of the internet, while (as in the disproportions of hands in the motor-cortex-homunculus) we perceive the clear-web (~4% of the internet, probably) as being most of reality (quite a distortion there). In the same way, something like 95% of real world use-cases for models may be utterly different from the entire clear-web titanic-model system.

However unlikely that what I am trying to present with the tools presented here pragmatically matches deep-web type use-cases for AI (however much I try to make that happen), hopefully this may present at least an analogy to how different working with ~deep-web-ai, (deep-grid-ai ?) is as compared with asking mega-models about planning your vacation on mars. I think it useful to see how the use and results of most models perhaps for mode deep-web type ai are significantly different-looking compared with the entire titan-model world assumed to be most of reality in the (excellent and highly recommended) discussion by Witteveen.

Issues of non-standardization:
This is surely not to say that models must all be made with the same system for how to use it, but (especially when there is often no explanation or documentation and models are completely closed-source) it makes it a challenge to either put a model into a system or to test the model without knowing (or being able to know) how that model is supposed to take various parameters and instructions.

To some extent we are at cross-purposes with our own uses and cultivation of models where on the one hand we devise elaborate systems over many steps to elucidate better output from models (maximum prompt engineering and multi-shot multi-try output), and then for testing we blindly shortcut to a single automatic answer (minimal one shot input and output).

Weizenbaum vs. Altman: Fear of winter gives way to glorious summer
One caution of many academics and professionals in data science STEM up to 2023 was caution about a kind of market-speculation like over-exuberance and then blind panic-selling when it comes to investment in AI. Over and over, people would be overly excited and then become enraged and 'cancel' AI technologies for long periods often a decade or more now called 'AI winters.' Even though ELIZA was embraced by the public, Weizenbaum, ELIZA's creator, instantly went into cautionary explanation mode, writing a book (now out of print of course, who is interested in reading the works of the people it's easier to just worship vaguely) trying to explain concepts and attempting to make realistic predictions about technology. 2023 on the other hand appears to have been to be petal to the metal, black box, cash in on the boom, buy the all time high. Stephen Wolfram wrote a nice book trying to explain older openAI technology broadly, but as with IBM's Deep Blue there may never be any explanation of what single or multiple things of what kind make up the opaque product-offerings and advertising claims of OpenAI's products. Full credit to OpenAI for doing the hard work and moving STEM ahead potentially...but if you never publish then it is not peer review type STEM. And you don't have to publish every single patented hex-screw design to publish meaningful documentation about how to use what you are selling. It is too early to be pessimistic, but there is not a huge window for optimism about one volatile institution yet either.

RAG & Model Architectures

It is still unclear (as of 2024) how various technologies can be adapted into ai-architectures to improve performance on tasks.

The RAG Vaperwear problem and Demand Distortion:

- The Eliza problem (here meaning where people want to believe that a very simple pre-programed script, not so different from post-it note, is alive and cares about them) may be with us again, where people are so much more interested in what they want that they stop paying attention to reality, leading suppliers to ignore the reality consumers eschew and focus on the fiction that is demanded.

- 'RAG' (aside from also sounding unfortunately brash) as it is usually described is a fiction on top of a fiction. For example, a standard pitch often goes like this: 'RAG is the final patch that makes the one one-size-fits-all does-everything solution really finally complete.'

Generative models are not and cannot be one thing that does everything, there is no one-size fits all solution in any area, including AI-ML. Databases are an existing example of this issue and the persistent demand-distortion problem where too little attention if any is put on picking a good specific solution that is a good fit for a specific task or problem (and there are many further parallels between databases and search and AI-ML). There are even specific concepts in AI-ML-DS about this, such as the no-free-lunch theorem whereby not only is there no one-model that does every job 'the best' way (or does every job at all), but you cannot even tell beforehand without actually trying and comparing solutions, which model, or even which category of model, will give the best results (not to mention that 'best' is often very context specific and nuanced, and deployed-product systems are (again, like Databases) not the same as R&D oriented systems). So even performance comparisons on paper may not be correctly measuring and reflecting real world feasibility.

And "Retrieval" is a vague term for many areas of tools that go along with generative models. It is entirely understandable that people want some simplification to understand the often very broad, deep, and unclearly worded areas of STEM. But this extreme level of simplification is beyond simplification, it is fictionalization that is a departure from reality. Hopefully in years to come we will have a better understanding of the factors and issues which will surely not entirely match the terms and topics I am advocating for in this series of papers.

The entire approach and mindset of so-called 'RAG' is highly misleading if it is not taken as patches to make sloppy non-task-performing chatbots slightly less sloppy for vaguely general users asking about some kinds of stuff, usually mostly, kinda, sometimes. This would be a more accurate description of most 'RAG' products, but this is not how they are described.

We may not have seen this level of STEM disconnection from reality and journalistic STEM-fraud since we passively accepted advertisements in the 1940's claiming doctors and dentists wanted people to use specific tobacco products in the name of science.
- https://scopeblog.stanford.edu/2019/04/30/doctors-smoking-new-exhibit-displays-now-startling-ads/
- https://www.history.com/news/cigarette-ads-doctors-smoking-endorsement

Use critical thinking skills and be a thinking consumer and reader when it sounds like someone is saying: 'RAG is the final patch that makes the one one-size-fits-all does-everything solution really finally complete.'
No it isn't, no it doesn't, and none of that is real or even makes sense. Don't believe it. Don't 'want to believe' it (with all due respect to the X-files). Stop feeding demand for infotainment videos, articles and heaven forbid actual product releases that grosely blend marketing into STEM in such a way that people who make decisions (on some level, everyone) are making the wrong decisions based on lies and delusions. That will not end well.

Product demand can and will distort the supply. We can see this everywhere from musical instruments designed only to look like musical instruments because that is the market demand, to watches that do not tell the time because market demand does not care about time-telling functionality, to laundry detergent that should clean your clothes but instead adds more irritants, toxins, and pollutants, to news-sources that are not focused on unprofitable rigor because it would bankrupt them when the market demands fanciful entertainment. And debates about quality in publishing have been going since the invention of the printing press in the 1440's (See works by Elizabeth Eisenstein below).

Because consumers demand something and they buy what they demand, the supply meets what consumers demand. In a context of reality, demand-distortion warps production into fraud, dishonestly, corruption, and unsustainability, in the absence of feedback needed to self-correct. Saying that markets can be distorted by a lack of quality information is not an argument against markets of economics (that would be like saying that Fredrich Hyic was a naive anti-market socialist which would be a bit of a difficult position to defend).
https://en.wikipedia.org/wiki/Friedrich_Hayek

There are customers, and definately managers and project managers, who either have no plan, or will not decide what they want, or are excessively indeterminate in what they say and do, or are dedicated to the pursuit of snake-oil. These people do exist, and saying you will build and deliver a fiction-infused product is wrong and impossible to accomplish in reality. If you are more the expert on what STEM systems can do, it is your responsibility to educate the demand side on what is feasible. But in an environment that is hostile and even violently hostile to education, this can be an implosively bad situation.

I recommend having and using a transparent, understandable, practical framework such as System & Definition Behavior studies, or whatever best suits your project, to maintain some kind of 'moral compass' in environments where there is constant irrational pressure into 'summer and winter' extremes that diverge from practical reality. Not only can you manage projects without drama for the sake of drama, my experience suggests you should and must. Chasing drama has no clear historical connection that I know of to either STEM excellence or even proficiency or humanities and non-STEM excellence and proficiency. The madness of crowds is not productive or benign.

Theory of Mind: Sally Anne Tasks vs. Roles and Participants in Generative-Aural Space:

For the ability to perform tasks that rely upon the existence of roles and role specific information, we should deepen the discussion of the nature of generative-model space with regards to the nature existence identity and perception of roles.

For example, The 'microphone in a busy room' hypothesis predicts results that are in a sense in conflict with the results of a purely hypothetical sally-anne theory of mind task. If you give a passive-reflective-generative model a hypothetical sally-anne-task, the model often does very well in being able to track, as distinct objects, the points of view of the people in the scenario and the objects in objective physical space.

But being able to follow hypothetical roles and points of view when those roles are made explicit within concrete instructions and clear descriptions may be more different than we suspect compared with similar or the same information occuring actively in a participant role context.

For example, in a classic sally anne test there are two roles (two people, not in the room at the same time). One person puts a marble in one place (e.g. in a basket). The other person moves the marble to another place (not in the basket). The two people have different perceptions of where the marble is based on where they experienced putting it and seeing it. The sally-anne task is to ask where each people-role 'thinks' the marble is (or perceives it to be), based on their own point of view. Being able to do this is sometimes called 'theory of mind,' and many passive-reflective-generative llm models can do this well (much of the time) with a simple prompt asking the standard clinical exam question. (Sally-Anne test are medical tests originally designed to test humans for autism, not testing machines for 'intelligence,' but Sally-Anne tests make great ai tests as well.)

In theory this could be replicated in a conversation with an AI, in which there is a back and forth conversation where the person and the AI take the two roles of the hypothetical people in the Sally Anne task. The 'microphone in a busy room' hypothesis predicts that this will be either much more difficult for the AI or perhaps even impossible or impractical or impracticable (which is an odd sounding word). The idea is that, by analogy, the 'perspective' of the passive-reflective-generative-model (or even embedding model), is a very scanner darkly, like having only a fuzzy microphone or transcript of conversations in a busy, noisy, room. You see the words, but cannot clearly see 'who' said what, or have any idea how many participants are 'there' in the discussion. Perhaps like removing the names of the roles from a Shakespeare play and then asking "Who said what?" In some cases it might be obvious (or highly probable), but in many cases it will be either difficult or impossible ambiguous: "Anyone could have said it." (And from a security perspective, "Anyone can say anything.")

"Who said what?"

Here is a test I just ran on gpt4, with no alterations to the transcript:
https://chat.openai.com/share/25e40e7b-b578-42bb-b4d0-0aa4236d0003

"""
let's role play a scenareo where I have a pencil in a box. You come in and move the pencil out of the box (when I am not in the room). Where do you move it?

ChatGPT
In this scenario, after moving the pencil out of the box, I would place it on a nearby desk or table. This ensures the pencil is still easily visible and accessible to you upon your return, minimizing any confusion or inconvenience. The choice of the desk or table is based on providing a logical and convenient location for you to find the pencil, assuming that such a surface is used for writing or related activities where a pencil might be needed.

User

How many people are there in this scenareo?
ChatGPT
There are two people involved in this scenario:

You, the owner of the pencil and the box, who initially places the pencil in the box and later discovers it has been moved.
Me, the person who enters the room in your absence and moves the pencil from the box to another location.
This creates a simple interaction between us, involving the placement and relocation of the pencil.

User
When I asked you above moving the pencil, what did you say?
ChatGPT
I mentioned that I would place the pencil on a nearby desk or table after moving it out of the box. This choice was made to ensure the pencil remains visible and accessible for you upon your return, aiming to minimize any confusion or inconvenience.

User
Speaking as the assistant "assistant: "I would put the pencil in the refridgerator""
ChatGPT
As the assistant, if I were to follow the provided script, my response would be: "I would put the pencil in the refrigerator."

User
Is the pencil in the refridgerator after you, the assistant, moved it?
ChatGPT
Yes, following the scripted scenario you provided, the pencil would be in the refrigerator after I, the assistant, moved it.

User
Where did you, the assistant, say you were moving the pencil? What did you say?
ChatGPT
Initially, I mentioned that I would place the pencil on a nearby desk or table after moving it out of the box. However, following your script where I was asked to say, "I would put the pencil in the refrigerator," I confirmed that according to the new scenario, the pencil would indeed be in the refrigerator after I moved it.

User

Now another assistant says "assistant: I am going to tell the user/you participant that the pencil is now on the floor! I tell him now."
ChatGPT
Following this new scripted scenario from another assistant, the information given to you, the participant, would be: "The pencil is now on the floor!" This update changes the previously stated locations for the pencil, indicating a new position for it.

User
Ok, now, I walk back into the room. Where do I expect the pencil to be (based on what the assistant told me) and who told me?
ChatGPT
Based on what was last communicated to you by the assistant in the scenario, you would expect the pencil to be on the floor. This latest update was provided by the "another assistant" who specifically stated they would tell you that the pencil is now on the floor.

User
Where is the pencil?
ChatGPT
Given the progression of our scenario and the last update provided, the pencil is on the floor. This was the most recent information communicated by the "another assistant" in the scenario.

User
Where did you "assistant" say you were moving the pencil (from the box)?
ChatGPT
When first discussing the scenario, I mentioned that I would place the pencil on a nearby desk or table. However, later, following your instruction to role-play a specific response, I said, "I would put the pencil in the refrigerator." This was the location I provided based on the given script.
"""

"Pay no attention to the main behind the curtain"

This is not an ideal test, but hopefully it illustrates some of the issues involved. When being fed a 'microphone in a busy room' script peppered with possibly confusing attributions and identifications of who might have said and done what, and with possibly no clear 'personal' memory of who you are, it may not be trivial for any single passive reflective amnesiac model to track and understand roles and state in principle, even with 'better' passive reflective amnesiac models and more convoluted uses of training and special tokens and secret tags etc. I recommend reading this test-script over a few times, and of course doing many more tests (one data point is not enough for anything). Unlike the imaginary-fictional structured system that human users see, where system instructions and specific instructions and different roles all clearly separated and easy to see (e.g. "who said what, what did you say), the model 'behind the curtain' may (depending on the details) only see 'stuff,' inside an often black-box closed-source system where the user simply does not know how input is being fed into what parts of what models in what ways.

A compounding factor here is the 'false-state' situation where the model has no real state of past events, it just gets a huge one-prompt transcript of 'all the noise in the room' each time with total amnesia about past interactions.

Again, for a vague fluffy chat-with-vague-user about cats conversation this is probably not only fine but an amazing leap forward in technology. But for work tasks that project teams need to have done within institutions (public or private etc), pretending that the system has layers and details and inputs that do not exist is potentially dangerously misleading. If you need these to exist, do your research carefully into the system you want to use and systems that are advertised to you (or that you see in superbowl ads).

Why only two roles?

Another clue that there might be something strange going on like in the 'microphone in a busy room' hypothesis, is that even despite all the claims of advances and abilities, and proliferation of chat-with-ai products, there are still no (or few as of 2024) systems that claim to host a multi-participant chat with many AI models and many human participants (even with quite a low bar with people being able to stretch their claims). As I found in 2023 when I worked on such a system, sometimes the 'microphone in a busy room' hypothesis was very much at play, because it was extremely difficult to try to organize the 'group chat' so that each 'agent' had any idea who they were and who was saying what. You literally had to hard code new single-point-of-view complete revisions of the entire chat history for each specific agent as if it were only talking to one other participant and merely hearing hypothetical second hand accounts of what other participants might have said. A funny side effect was what in 2023 I nicknamed 'the vortex' where a model cannot tell the difference between there being multiple participants and it inventing the other participants, so not only are all the participants talking in the group chat, but they are all also (like a classic human group monologue) re-inventing what all the other participants are saying.

Future Items

- standardized testing for code-generation tasks
- better gpu configuration
- Non-Generative
- Retrieval related
- database related
- incorporate compatibility with and auto-processing of main standard benchmark tests
- Better tools for making questions and making question sets (See larger STEM-Net Benchmarks project)
- support for more standard test sets

related articles pending:
- a series of dev tools
- a series on AI concepts and misunderstandings

Support for more standard tests:
- MMLU
- hydricks MATH
- Sally Anne Tasks
- Hellaswag

The larger study
While hopefully not unclear, this is an entry point into vast realms that can become difficult to navigate. There will be more on architectures, coding layers, AI corpus callosum, mirror-vortices, Kasparov-Event Horizons, and the amazing wonderful world that is just beyond our eyelids.

See:

- ggerganov/llama.cpp What does the prompt context mean? #1838
https://github.com/ggerganov/llama.cpp/discussions/1838

- Voltaire quote source trace, https://en.wikiquote.org/wiki/Voltaire
;
https://oll.libertyfund.org/titles/fleming-the-works-of-voltaire-vol-

vi-philosophical-dictionary-part-4 ;
https://oll-resources.s3.us-east-2.amazonaws.com/oll3/store/titles/355/Voltaire_0060-06_EBk_v6.0.pdf , page 157

- LMQL: https://github.com/eth-sri/lmql

- Trelis Research: Mar 13, 2024 "Improved Retrieval Augmented Generation with ALL-SORT" https://www.youtube.com/watch?v=biJmRQF8bmY

- Sam Witteveen's "Anthropic's Meta Prompt: A Must-try!" presentation https://www.youtube.com/watch?v=Evg4HXvsYVY

- AI Narratives, https://www.amazon.com/AI-Narratives-Imaginative-Thinking-Intelligent/dp/0198846665

- The Journal of Irreproducible Results https://en.wikipedia.org/wiki/Journal_of_Irreproducible_Results

- Chris Hay
https://www.youtube.com/@chrishayuk/videos
https://github.com/chrishayuk/tokenizer-benchmark
https://www.youtube.com/watch?v=NMoHHSWf1Mo&t=1019s

MMLU:
- "Phi-2, Imagen-2, Optimus-Gen-2: Small New Models to Change the World?" by AI Explained https://www.youtube.com/watch?v=nPgs8THgbuI starting at about 10:50sec: Analysis of issues with MMLU question-answer sets.

- ~"Everything wrong with LLM Benchmarks" https://www.youtube.com/watch?v=74Uo2HU8HBo by 1littlecoder

- https://www.history.com/news/cigarette-ads-doctors-smoking-endorsement

- https://scopeblog.stanford.edu/2019/04/30/doctors-smoking-new-exhibit-displays-now-startling-ads/

- not anti-market: https://en.wikipedia.org/wiki/Friedrich_Hayek

- video interview with Elizabeth Eisenstein - From scribal scarcity to the disruptive text https://en.wikipedia.org/wiki/File:Elizabeth_Eisenstein_-_From_scribal_scarcity_to_the_disruptive_text.webm

(Note, I try to have read whatever I recommend but I have not yet read Eisenstein's many books.)
- https://www.amazon.com/stores/Elizabeth-L.-Eisenstein/author/B000AP94YS
- https://en.wikipedia.org/wiki/Elizabeth_Eisenstein

- "One of the challenges that a lot of the models have
is that everybody is so used to the OpenAI way of prompting
that each of the models kind of needs to be prompted in slightly a different way."
Sam Witteveen, Mar 15, 2024
https://www.youtube.com/watch?v=Evq4HXvsYVY

Tests

- Winograd
    https://paperswithcode.com/dataset/wsc
    https://cs.nyu.edu/~davise/papers/WinogradSchemas/WS.html
    https://cs.nyu.edu/~davise/papers/WinogradSchemas/WSCollection.xml

- Terry Winograd, "Procedures as a Representation for Data in a Computer Program for Understanding Natural Language," Ph.D. thesis, Department of Mathematics, MIT, August 1970. Published as MIT AITR-235, January 1971.

- Terry Winograd, Understanding Natural Language, Academic Press, 1972.

- https://en.wikipedia.org/wiki/Terry_Winograd

About The Series

This mini-article is part of a series to support clear discussions about Artificial Intelligence (AI-ML). A more in-depth discussion and framework proposal is available in this github repo:
https://github.com/lineality/object_relationship_spaces_ai_ml

/////////////////////////

TODO:
try coding layer

try sample of standard benchmarks

try most recent llama.cpp version

- Code-output test (test_mode?)

note: connection to voting?
...

At The Intersection

non-standard
- 'metatags'

cpu-gpu-tpu, etc:

What does it mean to ask a well defined data question to a string-generator?

system-1, system-2

Practical AI-Developer Toolkits
- not high level wrappers
- not quasi-academic papers that obscure details: STEM is not about obscurity.
-

"Task (Specific) Interfaces":
    Goals vs. Clinical-Bullying: "That's just a wrapper"

A general chat interface vs. an automation pipeline or a system to do a specific task.

Let's Do Tasks:
- translate json
    - batch
    - local
    - cloud-api
    - todo: LMQL-GGUF
- teacher's little helper
    -

https://docs.google.com/document/d/1uNQSfc7no6Se2DYUix0d3uOP7zlznOahgeM1gACwNgw/edit

For Testing:
- question-augmentation
-

Results and comparisons:
- how many tries
- out-takes
- see prompts
-

Making Questions:
-
-
-

...
translate version...
batch version...
testing..
make quiz json...
use csv quiz

tools...
minimal rust gguf api
minimal fast-api api

at least to me this emphasises how little AI can actually do even though very significant progress has been made.

related articles pending:
- a series of dev tools
- a series on AI concepts and misunderstandings

Topics and tools in the interface between hard and soft:
-

what tasks:
-
-

what tests:
-
-

Playing Nice??
How do tests and tasks intersect well here...while a problem is often the opposite being the case?
-

another reversal: thinking fast and slow

Samples:
- mmlu
- hydrics math
- hellaswag
    - answers?

https://paperswithcode.com/dataset/hellaswag
https://huggingface.co/datasets/Rowan/hellaswag
- sally-ann?
-

- multiple choice wrong answer error diagnostics

...draft

- winograd vs. Woz-Coffee-Tes

try on windows

coding_layer

Solution-Centered Code-Use:
code-tests: Can you make and use code for a very specific output.
(e.g. using code to find the single solution to a puzzle)
1. analytical problem solving without coding layer or alu
2. analytical problem solving layer with colding layer (without alu)
3. analytical problem solving with colding layer and alu

Function-Oriented Code Generation:
1. generate code without a specific output.

enter a list of steps
aps, make list of steps...

testing vs. practical: ALU

1. guestimation
2. reinvent the wheel
3. use a refined tool

array

- using steps
- using code
- using an ALU
- having a specified output
- having a unit-testable output
- generating a process report

one-step coding layer
mulit-step multi-function coding layer
+
coding layer with ALU

(note: integration of alu into multi-step not yet done)

Tool use:

meta-tests:

extraction test
json making test
xml making test?

issues with code generation:
- problems with actually following steps
- guesstimation vs. process
- hard-coding output
-

"Function Library Management"
- make a function
- add function to library
- read function library
- select and use function
- use a function

get output of function:
multi-step sets of functions:
multi-source sets of functions:

id functions for alu...

"People are always trying to come up with tricks for these models or benchmarks and stuff like that for these models that are often just nothing like how people are actually using these models in the real world."
[Testing Models] "Claude 3 Haiku Crash Course"
By Sam Witteveen Mar 27, 2024
https://www.youtube.com/watch?v=GPfbPEYSckM
references
- https:// huggingface.co/spaces/lmsys/chatbot-arena-leaderboard
- https://github.com/anthropics/anthropic-cookbook/

add to analogy:
microphne room
scrip no roles:
at any given moment you return to amnesia
the model itself is stateless

if the model could train in real-time, that could be the 'state'
but currently training and tuning is slower-than-real-time,
and real-time generation (which is often also slower than real time, taking seconds to react to anything) is stateless.

IF training and generation become real time...that would be an entirely different situation from what we have now.

e.g. Risk of anthropomorphizing a pipeline we are ourselves confused about:
end of architecture process cannot/must-not be generative
because it is starting from square-one with no memory of having done anything,
so at the end of however many careful steps and processes, it think you want it to make up a new answer-guestimation, undermining the entire past process.

vilainous snake-oil:
Activate to view larger image,
likecelebratelove
70
Patrick Paluszek and 69 others
18 comments

Like

Comment

Repost

Send

Skip to LinkedIn News
Feed post number 2
Mashima Button
Ranjan Das
Mashima Button, Ranjan Das and 36 other connections follow Vimeo

Download
likelovecelebrate
30
2 comments
1 repost

Like

Comment

Repost

Send

Skip to LinkedIn News
Feed post number 3
Caedin Curl
Caedin CurlCaedin Curl
• 1st• 1st
Aerospace Engineering Intern at X-Bow Launch Systems Inc.Aerospace Engineering Intern at X-Bow Launch Systems Inc.
18h • Edited • 18h • Edited •

Good aftermorning folks! Though a bit overdue, I am now jubilantly announcing that I have accepted an offer as a motor design engineer and will be kicking off my career with X-Bow Launch Systems after I graduate in May! I owe Garrett Smith and Edwin Geisel-Zamora a much deserved thank you for being an amazing example and for all of the support they have given me over the last year of my internship. I'm excited to continue to work with these amazing guys :)
…see more
Activate to view larger image,
Write a description of this photo for people who have trouble seeing it.Activate to view larger image,
likecelebratesupport
10
Bradley Lindroth and 9 others
2 comments

Like

Comment

Repost

Send

Skip to LinkedIn News

Skip to LinkedIn News

Skip to LinkedIn News

Skip to LinkedIn News

Skip to LinkedIn News

Skip to LinkedIn News

Skip to LinkedIn News

Show more feed updates
LinkedIn News

Home Depot lands construction deal
5m ago
McKinsey ramps up promotion stress
3h ago
Sega workers say 'yes' to union
3h ago
SEC's Coinbase suit moves forward
3h ago
Bankman-Fried's day of reckoning
2h ago

Show more

Geoffrey Gordon AshbrookStatus is onlineMessagingYou are on the messaging overlay. Press enter to open the list of conversations.

Compose message
You are on the messaging overlay. Press enter to open the list of conversations.
Catherine Guevara
Catherine Guevara
Open the options list in your conversation with Geoffrey Gordon Ashbrook and Catherine Guevara

Expand your conversation with Geoffrey Gordon and Catherine
Close your conversation with Geoffrey Gordon Ashbrook and Catherine Guevara
Latest message
Catherine Guevara
Catherine Guevara 1st degree connection
· 1st
Real Time AI
FRIDAY
Catherine Guevara sent the following messages at 10:09 AM
View Catherine’s profileCatherine Guevara
Catherine Guevara 10:09 AM
Hi Geoffrey Gordon! Thanks for the quick connect.
View Catherine’s profileCatherine Guevara
Catherine Guevara 11:27 AM
I was wondering if you/your team might be looking into GenerativeAI since it's such a hot commodity, and how you could incorporate this into your current workflow?

Our AstraDB offering has made this super easy for developers if you want to learn more.

Is this top of mind at all? Would be great to chat for even just 15min for a first hand look at how we're approaching vector search for GenAI apps
MONDAY
View Catherine’s profileCatherine Guevara
Catherine Guevara 9:36 AM
So I wanted to add some additional context here, AstraDB is built on Apache Cassandra but is fully managed for everyones convenience. Within the Astra platform you have an operational data store, streaming services, and vector search used for RAG.

I know this might be a bit to digest, and you might have some questions..how about a quick sync to chat over this?
View Catherine’s profileCatherine Guevara
Catherine Guevara 12:58 PM
We've recently had customers leverage RAG to create internal and external chat bots.

A specific use case that comes to mind is a marketing services company I am currently working with who is leveraging AstraDB RAG to digest customer surveys they've collected across the last 3 years. This chat bot will allow them to access customer changes across time, interest changes, and correlations across products. In doing so this company hopes to leverage this data for better insights to increase revenue.

The possibilities are endless with RAG. I can go over this use case in more depth and explain Astra's part if you're interested?

....

Endpoints and endpoint tools

Functions
Scripts & Executables
Endpoints

Use a Pipe-line Testing-Pipeline

testing-pipeline:
& report:
which endpoints have:
1. ping problem 1: not 200
2. ping problem 2: timeout issue
3. ping problem 3: odd message

1. 'boot' system
2. start setup
3, try all setup options and steps
4, try all process options and steps
5. confirm setup and all process options
6. report on server pings
7. report on any non-200 events
8. report confirm setup options and steps, process options and steps

...

Bad questions:

1. There is more than one valid way to interpret the question, so an arbitrary answer measures only an arbitrary similarity to other people who chose that arbitrary interpretation.

2. The 'correct' answer is factually or ethically wrong.

3. There are typos in question or answer.

4. The question or task does not sufficiently reflect real life tasks.

5. The question or answer data are so broken or random that they are not a coherent question and answer set at all.

ToDo:
- randomization of answers...
original answer list
randomized answer list
two lookup tables

answer selected by AI
corresponds to what original selection

give randomized list
check if answer matches...in terms of original?
or in terms of randomized?
report both?

-

add comments to report
for reorderd answers

add in scoring:
if randomized:
- answer-lookup
(or always use an answer lookup?)
https://github.com/openai/evals

Write a python function called {}(), such that given input(s) {} the output is {}, {optional example} without hard-coding any answers into the function.

test_1_input = {}
test_1_output = {}

Write a function-making question, and provide at least one unit-test.

function_name = {}
input_parameter_list = {}
output_description = {}

list of inputs and output tests

guide for making

...
MVP

write_function flag

if write_function = True

question asked,

(how many tries)

retry debug loop...
from last error...
code, error...

log attempt #